Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Informatica

# L'ALGORITMO DI DEUTSCH-JOZSA IN QSCRIPT

**Relatore:**
**Chiar.mo Prof.**
**UGO DAL LAGO**

**Presentata da:**
**FEDERICO PECONI**

**II Sessione**
**a.a. 2016/2017**

*Alla mia famiglia, che mi ha regalato un cuscino su cui cadere.*
*A Bologna, che mi ha insegnato a respirare più profondamente.*
*A tutte le persone che mi hanno accompagnato in questi anni.*
*Dedico a voi questo lavoro.*
*408/2*

# Indice

# Capitolo 1

# Introduzione

Che cosa hanno in comune: l'analisi di un mercato finanziario in continua crescita, la modellizzazione delle interazioni a scala molecolare, la simulazione dei cambiamenti climatici e la crittoanalisi dei moderni sistemi crittografici che garantiscono la sicurezza di internet? Per quanto apparentemente distanti, questi problemi sono legati tra di loro da un denominatore comune: per essere portati a termine necessitano di enormi quantità di calcoli. Sia che emerga con la valutazione dei risultati di grandi quantità di previsioni finanziarie o che si manifesti nella ricerca dei fattori primi di un numero molto grande per decifrare RSA [22], la complessità computazionale derivata da questi problemi rappresenta un ostacolo difficile da superare anche dai moderni super computer. In realtà, per il calcolo digitale, questi problemi non sono solo attualmente intrattabili ma, con ogni probabilità, lo saranno per sempre. Ciò è dovuto al numero di operazioni necessarie per risolverli, che cresce, in relazione con la dimensione del problema, con ritmo esponenziale, almeno se concentriamo la nostra attenzione sugli algoritmi esistenti. A titolo di esempio, la simulazione delle interazioni di piccole molecole composte da pochi atomi può essere portata avanti anche da un laptop domestico nell'arco di qualche secondo mentre, aumentando in ordini di grandezza e passando a molecole composte da alcune centinaia di atomi, il processo di esatta simulazione impegnerebbe un qualsiasi computer per un tempo superiore a quello

dell'età dell'universo.

## 1.1 Il Paradigma Quantistico

La buona notizia è che una soluzione sembra esistere e deriva da un approccio drasticamente differente rispetto a quello tradizionale. Il modello di calcolo classico sfruttato dai moderni calcolatori, infatti, si fonda sul il concetto di bit: un'unità logica che può avere valore 0 o 1 e sulle operazioni effettuabili su questi valori. È sufficiente questo modello con l'aggiunta di poche altre regole che verranno chiarite in seguito per derivarne l'Informatica come oggi la conosciamo, con tutte le sue applicazioni e limitazioni, alcune delle quali appena esposte. Essenzialmente, dai tempi di Turing, nessun cambiamento ha avuto luogo tale da mettere in discussione che cosa sia e come opera un computer fino a quando, dopo l'affermazione della meccanica quantistica, negli anni 80 del secolo scorso, alcuni scienziati cominciarono ad avanzare l'ipotesi che le stesse leggi che governano il mondo infinitamente piccolo potessero essere sfruttate per lo sviluppo di un nuovo paradigma di calcolo dalle nuove potenzialità [6]. La proprietà che veniva proposta della meccanica quantistica come più rilevante ai fini dell'applicazione alla computazione era il postulato secondo il quale un sistema fisico quantistico con $n$ gradi di libertà si può trovare contemporaneamente in un numero di stati che cresce esponenzialmente con $n$. Come ogni struttura fisica anche gli stati di un sistema quantistico possono essere utilizzati per trasportare informazione e quindi aprire la strada ad un'elaborazione a sua volta esponenziale.

Analogalmente, anche un computer quantistico possiede un'unità di base che è chiamata quantum bit o, in maniera abbreviata, qubit. Come anticipato e come verrà poi meglio formalizzato, il qubit si distingue dal bit classico per la possibilità di trovarsi in una *sovrapposizione* dei valori logici 0 e 1 contemporaneamente oltre che assumere i valori 0 e 1 proprio come un bit. Nella pratica un qubit sarebbe implementatbile sfruttando le proprietà tipicamente quantistiche quali la polarizzazione di un fotone o lo spin di un elettrone; lo

studio delle tecniche e i metodi per la realizzazione di un computer quantistico richiede avanzate conoscenze di fisica ed esula dai fini della tesi in oggetto. Qui basti accennare che le difficoltà tecniche in questo campo sono ancora tante e, allo stato dell'arte, solo poche compagnie dichiarano d'aver sviluppato "piccoli" chip quantistici funzionanti [24].

In questo documento vengono trattate da un punto di vista teorico le fondamenta del calcolo quantistico, ponendo l'attenzione in particolare sulla progettazione di algoritmi che sfruttino il parallelismo emergente dalla sovrapposizione, andando infine ad analizzare un preciso caso d'uso.

## 1.2 Potenziale

La fattorizzazione di numeri interi in numeri primi è, contrariamente a quanto avviene per il calcolo tradizionale, un problema relativamente facile da risolvere da un computer quantistico. Peter Shor fornì nel 1999 un algoritmo quantistico che richiede tempo polinomiale per risolvere il problema [25]. Questo algoritmo è, teoricamente, in grado di rompere i sistemi di sicurezza a chiave pubblica utilizzati ad esempio in molte transizioni bancarie. D'altra parte sono stati sviluppati protocolli a crittografia quantistica per lo scambio di chiavi teoricamente immuni a qualsiasi tipo di attacco, si parla in questo caso di crittografia perfetta [26]. Al di fuori della fattorizzazione, gli algoritmi quantistici risultano essere ideali nella simulazione di processi fisici a stato solido e chimici vantando sempre una diminuzione più che polinomiale in tempo rispetto alle rispettive procedure classiche [27]. Esistono problemi per cui i computer quantistici garantirebbero una diminuzione polinomiale in tempo: l'esempio più famoso è dato dall'algoritmo di Groover [28] per la ricerca di un record all'interno di un database, che assicura una riduzione quadratica delle richieste per trovare l'elemento cercato.

Si ritiene che le simulazioni quantistiche permetterebbero di modellare fedelmente il comportamento di atomi in condizioni inusuali come ad esempio durante gli scontri all'interno degli accellaratori di particelle [29] e, in gene-

rale, molti ritengono la simulazione il campo ideale in cui la computazione quantistica risulterebbe vincente [30], rispondendo così alle domande con cui si è aperta la trattazione.

Lo spettro delle applicazioni in cui l'informatica quantistica garantirebbe vantaggi è quindi vasto e promettente ed è correntemente supportato da numerosi investimenti che fanno di questa disciplina uno dei campi di ricerca più avanzati ed interessanti dell'Informatica.

Introdotte le motivazioni che difendono l'importanza d'approfondire questo campo, viene di seguito fatta una panoramica della struttura che questa tesi seguirà.

## 1.3 Di Che Cosa Tratta Questa Tesi?

L'intera tesi ha come punto d'arrivo l'analisi della correttezza e l'implementazione di un algoritmo quantistico : l'algoritmo di Deutsch-Jozsa. L'algoritmo di Deutsch-Jozsa, come vedremo, non risolve un problema che ha dirette applicazioni pratiche bensì è un risultato importante dal punto di vista teorico. Fu infatti il primo esempio di procedura quantistica (deterministica) in grado di ottenere una diminuzione esponenziale in tempo rispetto alla controparte classica per la risoluzione di un problema.

Per arrivare a parlare di Deutsch–Jozsa è necessario acquisire i concetti fondamentali che caratterizzano il calcolo dei quanti, a tal proposito si è scelto di sviluppare la trattazione in capitoli e sezioni, ognuna incaricata di esporre al lettore una specifica tematica al fine di guidarlo verso la piena comprensione della tesi. In particolare:

- nel secondo capitolo vengono trattati gli aspetti generali delle funzioni Booleane che sono alla base dell'algebra di ogni computer concentrando l'attenzione sulle funzioni Booleane bilanciate, una particolare classe che tornerà poi utile nell'ultimo capitolo;

- nel terzo e principale capitolo invece si comincia introducendo quella che è stata la storia che ha portato alla nascita dell'informazione quan-

tistica, elencandone i protagonisti e le scoperte principali. Vengono poi informalmente esposte le leggi e le stranezze della meccanica quantistica. Dopo questa parte propedeutica sono definiti e messi a confronto, in ordine, il sistema di calcolo classico, il sistema probabilistico [1] ed infine quello quantistico. Si passa poi a ricavare la dinamica del sistema e generalizzarlo su più qubit per mostrare le potenzialità offerte dal parallelismo che ne deriva. Viene inoltre aperta una parentesi sul fenomeno dell'entanglement e la sua importanza all'interno del campo e infine presentato il primo esempio concreto di algoritmo quantistico;

- il quarto capitolo è incentrato sull'analisi della correttezza dell'algoritmo di Deutsch–Jozsa e sulla formulazione del problema da questo risolto. Viene come conseguenza quantificato lo scarto di complessità nei confronti delle relative procedure classiche;

- il quinto e ultimo capitolo consiste nel riportare il processo di progettazione ed esecuzione seguito per l'implementazione di Deutsch–Jozsa. Viene utilizzato un simulatore quantistico fornito da un'applicazione web e un linguaggio di scripting per codificare le varie istanze. Vengono ripresi i risultati del primo capitolo come test su cui effettuare l'esecuzione.

[1] Il modello probabilistico qui esposto è adattato per meglio introdurre il modello quantistico

# Capitolo 2

# Funzioni Booleane Bilanciate

Come già anticipato nell'Introduzione, le funzioni Booleane bilanciate sono una struttura matematica su cui poggia parte del contenuto di questa tesi. È risultato perciò utile, ai fini di una trattazione chiara ed esaustiva, impiegare un capitolo per definirne in maniera rigorosa i concetti di base.
"*Per fare un tavolo ci vuole il legno*" recita l'inizio di una famosa canzone per bambini, ad indicare la natura celata delle cose che, così nella quotidianità come nella matematica, spesso necessitano di altre conoscenze per essere comprese appieno; seguendo quindi questa impronta fondazionale, andiamo per prima cosa ad introdurre le funzioni Booleane.

## 2.1 Funzioni Booleane

Le funzioni Booleane apparvero per la prima volta a metà diciannovesimo secolo durante la formulazione matematica di problemi logici e prendono il loro nome da George Boole, matematico britannico considerato fondatore della logica matematica odierna [2].

Una semplice funzione Booleana può può per esempio $f : \{0,1\}^2 \to \{0,1\}$ definita come segue

$$f(00) = 0$$
$$f(01) = 0$$
$$f(10) = 1$$
$$f(11) = 0$$

oppure da $g : \{TRUE, FALSE\} \to \{TRUE, FALSE\}$

$$g(FALSE) = TRUE$$
$$g(TRUE) = FALSE$$

Notiamo come entrambe abbiano in comune la dimensione del codominio e la capacità di agire su un numero finito di valori appartenenti ad un insieme di 2 elementi. È tuttavia conveniente operare su di un insieme che possa essere visto sia dal punto di vista qualitativo (vero, falso) sia da un punto di vista quantitativo, e quindi numerico, che ci permetta così di compiere anche operazioni algebriche oltre che logiche. Prediligeremo allora da qui in avanti l'insieme $\{0,1\}$ come campo vettoriale su cui lavorare.

**Definizione 2.1.** *Una* funzione Booleana a $n$ variabili *è una funzione da* $\mathcal{B}^n$ *a* $\mathcal{B}$*, dove* $\mathcal{B} = \{0,1\}$*,* $n > 0$ *e* $\mathcal{B}^n$ *è l'n-esimo prodotto cartesiano di* $\mathcal{B}$ *con se stesso [3].*

**Corollario.** $\forall\ n > 0$*, ci sono* $2^{2^n}$ *funzioni da* $\mathcal{B}^n$ *a* $\mathcal{B}$*.*

*Dimostrazione.* Sia $\mathcal{F} = \{f | f : \mathcal{B}^n \to \mathcal{B}\}$, ogni $f$ riceve in input n-uple $\vec{x} = (x_1, .., x_n)$ che possono essere viste come sequenze di $n$ bit. In $n$ bit possiamo codificare, trattandosi di una distribuzione con ripetizione di classe $n$, $2^n$ oggetti differenti, quindi $\mathbb{D}_{2,n} = |\mathcal{B}^n| = 2^n$. Per definizione di $f$, per ogni $\vec{x}$, $f(\vec{x}) = 0$ oppure $f(\vec{x}) = 1$, quindi ogni possibile $f$ individua un sottoinsieme di $\mathcal{B}^n$. Allora $\mathcal{F}$ avrà cardinalità uguale all'insieme delle parti per $\mathcal{B}^n$, quindi $|\mathcal{F}| = \left|2^{\mathcal{B}^n}\right| = 2^{|\mathcal{B}^n|} = 2^{2^n}$ □

Un altro modo più tradizionale per descrivere una funzione Booleana è quello di fornire la sua tabella di verità. Ad esempio, per la $f$ precedentemente definita, la relativa tabella di verità sarà:

| $x_1$ | $x_2$ | $f(x_1, x_2)$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

dove a destra viene posto il risultato della funzione calcolata sui valori delle colonne precedenti.

In entrambe le rappresentazioni, tuttavia, le funzioni vengono descritte in maniera implicita mostrando solamente input ed output, senza mai andare a specificare come questo output venga calcolato. Nell'ultimo capitolo, per l'implementazione dell'algoritmo di Deutsch-Jozsa, verranno utilizzate funzioni Booleane *bilanciate* che necessitano di essere calcolate esplicitamente. La prossima sezione introduce questa categoria di funzioni Booleane e propone due metodi effettivi e generali per produrne istanze concrete.

## 2.2 Classi di Funzioni Booleane Bilanciate

Sia $f$ una funzione Booleana, chiamiamo $\vec{x} = (x_1, \cdots, x_n)$ *vettore positivo* per $f$ se e solo se $f(\vec{x}) = 1$, *vettore negativo* altrimenti. Sia $(|f|^1, |f|^0)$ la partizione dove $|f|^1$ è la quantità che indica il numero di vettori positivi per $f$ e $|f|^0$ la quantità che indica il numero di vettori negativi.

**Definizione 2.2.** *Una funzione Booleana $f$ è detta bilanciata (FBB) se e solo se $|f|^1 = |f|^0$.*

**Corollario.** $\forall\ n > 0$, *ci sono* $\frac{2^n!}{(2^{n-1}!)^2}$ *FBB da* $\mathcal{B}^n$ *a* $\mathcal{B}$.

*Dimostrazione.* Per definizione, una FBB $fbb : \mathcal{B}^n \to \mathcal{B}$ è individuata da una partizione $(\mathcal{M}, \mathcal{N})$ con $\mathcal{M} \subset \mathcal{B}^n$ e $\mathcal{N} \subset \mathcal{B}^n$ tale che $|\mathcal{M}| = |\mathcal{N}| = 2^{n-1}$

e $fbb(m) = 0 \wedge fbb(n) = 1 \; \forall m, n$ con $m \in \mathcal{M}, n \in \mathcal{N}$. Quindi, il numero delle possibili funzioni corrisponde a tutti i possibili modi di partizionare a metà un insieme di $2^n$ elementi e, a sua volta, ogni partizione così creata può essere rappresentata da un sottoinsieme $\mathcal{S} \subset \mathcal{B}^n$ dove $\mathcal{M} = \mathcal{S}$ e $\mathcal{N} = \mathcal{B}^n \setminus \mathcal{S}$. Non resta quindi che trovare tutti i possibili $\mathcal{S}$:

$$\mathbb{C}_{2^n,2^{n-1}} = \binom{2^n}{2^{n-1}} = \frac{2^n!}{2^{n-1}!2^{n-1}!} = \frac{2^n!}{(2^{n-1}!)^2}$$

□

Una valida tabella di verità per una FBB in $\mathcal{B}^3$ portebbe essere la seguente:

| $(x_1, x_2, x_3)$ | $g(x_1, x_2, x_3)$ |
|---|---|
| $(0, 0, 0)$ | 0 |
| $(0, 0, 1)$ | 1 |
| $(0, 1, 0)$ | 1 |
| $(0, 1, 1)$ | 0 |
| $(1, 0, 0)$ | 1 |
| $(1, 0, 1)$ | 0 |
| $(1, 1, 0)$ | 0 |
| $(1, 1, 1)$ | 1 |

Dove, giustamente, si noti come il numero dei vettori positivi equivale al numero dei vettori negativi.

Ma come è costruita $g$ nello specifico? Qual'è la computazione che mi porta a determinati outputs? Per rispondere a queste domande si può provare a cercare qualche correlazione tra i valori in entrata e i valori in uscita: dopo un pò di riflessione dovrebbe saltare all'occhio che: ogni qual volta abbiamo un numero dispari di 1 nell'input la funzione restituisce 1, restituisce invece 0 quando tale numero è pari. La parità di una sequenza binaria è esprimibile con la somma modulo 2 dei singoli componenti, allora possiamo

definire esplicitamente $g$ come:

$$\begin{aligned} g(x_1, x_2, x_3) =& x_1 + x_2 + x_3 \mod 2 \\ =& (x_1 \oplus x_2) \oplus x_3 \\ =& x_1 \oplus x_2 \oplus x_3 \end{aligned}$$

dove $\oplus$ è un connettivo binario logico (una funzione Booleana $\in \mathcal{B}^2$) chiamato XOR oppure OR esclusivo, che restituisce 1 se uno ed uno solo dei due elementi vale 1, implementando così l'operazione di modulo 2.

| $(x_1, x_2)$ | $\oplus(x_1, x_2)$ |
|---|---|
| $(0,0)$ | 0 |
| $(0,1)$ | 1 |
| $(1,0)$ | 1 |
| $(1,1)$ | 0 |

**Lemma 1.** *Per ogni $n > 0$ la funzione Booleana $f^n(x_1, x_2, \cdots, x_n) = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ è una FBB.*

*Dimostrazione. (per induzione su n)*

**Base induttiva:** $f^1(x) = x$ è banalmente bilanciata.

**Ipotesi induttiva:** Ipotizziamo $f^n$ bilanciata, quindi esiste la partizione $(\mathcal{M}, \mathcal{N})$ con $\mathcal{M} \subset \mathcal{B}^n$ e $\mathcal{N} \subset \mathcal{B}^n$ tale che:
$|\mathcal{M}| = |\mathcal{N}|$, $\mathcal{M} \cup \mathcal{N} = \mathcal{B}^n$ e $\forall m, n$ con $m \in \mathcal{M}$,$n \in \mathcal{N}$ risulta $f^n(m) = 0$ e $f^n(n) = 1$.

**Passo induttivo:** Avremo $f^{n+1} : \mathcal{B}^{n+1} \to \mathcal{B}$ dove $\mathcal{B}^{n+1} = \mathcal{B}^n \cdot 0 \cup \mathcal{B}^n \cdot 1$ e quindi
$|\mathcal{B}^{n+1}| = 2\,|\mathcal{B}^n|$. Dobbiamo dimostrare che esiste una partizione $(\mathcal{M}', \mathcal{N}')$ t.c.:

- $|\mathcal{M}'| = |\mathcal{B}^n|$ e $\forall m' \in \mathcal{M}'$ $f^{n+1}(m') = 0$
- $|\mathcal{N}'| = |\mathcal{B}^n|$ e $\forall n' \in \mathcal{N}'$ $f^{n+1}(n') = 1$

Notiamo come, $\forall m, n$ con $m \in \mathcal{M}, n \in \mathcal{N}$ se
$x_{n+1} = 0 \Rightarrow f^{n+1}(m \cdot 0) = 0 \wedge f^{n+1}(n \cdot 0) = 1$
$x_{n+1} = 1 \Rightarrow f^{n+1}(m \cdot 1) = 1 \wedge f^{n+1}(n \cdot 1) = 0$
Segue naturalmente che la partizione $\mathcal{M}' = \mathcal{M} \cdot 0 \cup \mathcal{N} \cdot 1$ e $\mathcal{N}' = \mathcal{M} \cdot 1 \cup \mathcal{N} \cdot 0$ rispetta i criteri cercati.

□

Si richiami dall'Algebra Lineare che, $f : \mathcal{V} \to \mathcal{W}$ è un'applicazione lineare se e solo se esiste una matrice $M \in \mathbb{M}^{k \times l}$ tale che $f(\vec{x}) = M\vec{x}$, dove $k = |\mathcal{V}|$ e $l = |\mathcal{W}|$ sono le dimensioni degli spazi vettoriali su cui agisce.

**Definizione 2.3.** *$b : \mathcal{B}^n \to \mathcal{B}$ è una funzione booleana lineare (FBL) se e solo se esiste $M \in \mathbb{B}^{n \times 1}$ tale che $b(\vec{x}) = M\vec{x} = c_1x_1 + c_2x_2 +, \cdots, +c_nx_n mod 2 = (c_1 \wedge x_1) \oplus (c_2 \wedge x_2) \oplus, \cdots, \oplus(c_n \wedge x_n)$. Dove $\oplus$ è l'operatore di somma per $\mathcal{B}^n$.*

**Corollario.** *Per ogni $n > 0$, $f^n$ è una FBL.*

*Dimostrazione.* Triviale, basta infatti porre $M = \overbrace{(1, 1, \cdots, 1)}^{n-volte}$. □

### 2.2.1 La Classe delle Funzioni Booleane Lineari

Introdotte le FBB e le FLB, possiamo a questo punto procedere con la formulazione del seguente risultato generale:

**Teorema 2.1.** *Sia $\mathbb{FL} = FLB \setminus \{f | f : \mathcal{B}^n \to 0, \forall n > 0\}$ la classe di tutte le funzioni Booleane lineari che non siano costanti a 0. Allora, $\forall f \in \mathbb{FL}$ $f$ è bilanciata.*

*Dimostrazione. (per induzione su n)*
Sia $l^n$ una qualsiasi funzione da $\mathcal{B}^n$ appartenente a $\mathbb{FL}$. Vogliamo dimostrare che $l^n$ è bilanciata.

**Base induttiva:** $l^1(x) = x$ è banalmente bilanciata.

**Ipotesi induttiva:** Ipotizziamo $l^n$ bilanciata.

Esisterà quindi la partizione $(\mathcal{M},\mathcal{N})$ con $\mathcal{M} \subset \mathcal{B}^n$ e $\mathcal{N} \subset \mathcal{B}^n$ tale che: $|\mathcal{M}| = |\mathcal{N}|$, $\mathcal{M} \cup \mathcal{N} = \mathcal{B}^n$ e $\forall m, n$ con $m \in \mathcal{M}$,$n \in \mathcal{N}$ risulta $l^n(m) = 0$ e $l^n(n) = 1$.

**Passo induttivo:** Si ricordi che $l^n$ può utilizzare tutti gli input ed avere la forma $l^n(x_1, x_2, \cdots, x_n) = x_1 \oplus x_2 \cdots \oplus x_n$ ma può anche avere la forma $l^n(x_1, x_2, \cdots, x_n) = x_3 \oplus x_4 \oplus x_8$ e non dipendere da $n$. Essendo una qualsiasi funzione lineare Booleana di dimensione $n$, in generale, a seconda della $M$ associata, $l^n$ nel suo corpo può utilizzare un qualsiasi sottoinsieme dell'input. Allora segue che $l^{n+1}$ può assumere esclusivamente una delle seguenti forme:

- **Singoletto** $\Rightarrow l^{n+1}(\vec{x}) = l^{n+1}(x_1, x_2, \cdots, x_{n+1}) = x_{n+1}$
  Che è bilanciata in quanto, sia $(\mathcal{M},\mathcal{N})$ la partizione dove $\mathcal{M}$ è composto da tutte le stringhe binarie $\in \mathcal{B}^{n+1}$ che codificano in base due i numeri da 0 a $2^n - 1$ e $\mathcal{N}$ composto da tutte le stringhe binarie $\in \mathcal{B}^{n+1}$ che codificano in base due i numeri da $2^n$ a $2^{n+1}-1$ i.e. :
  $\mathcal{M} = \{\vec{x} \in \mathcal{B}^{n+1} | \vec{x} = 0 \cdot x_n \cdot x_{n-1} \cdots x_1\}$
  $\mathcal{N} = \{\vec{x} \in \mathcal{B}^{n+1} | \vec{x} = 1 \cdot x_n \cdot x_{n-1} \cdots x_1\}$
  Allora $\mathcal{M} \subset \mathcal{B}^{n+1}$ e $\mathcal{N} \subset \mathcal{B}^{n+1}$, $|\mathcal{M}| = |\mathcal{N}|$, $\mathcal{M} \cup \mathcal{N} = \mathcal{B}^{n+1}$ e e $\forall m, n.\ m \in \mathcal{M}$,$n \in \mathcal{N}$ risulta $l^{n+1}(m) = 0$ e $l^{n+1}(n) = 1$.
- **Uguale a** $l^n \Rightarrow l^{n+1}(x_1, x_2, \cdots, x_{n+1}) = l^n$ e quindi bilanciata per ipotesi.
- **Xor di** $l^n \Rightarrow l^{n+1}(\vec{x}) = l^{n+1}(x_1, x2, \cdots, x_{n+1}) = x_{n+1} \oplus l^n$.
  E quindi se:
  $x_{n+1} = 0 \Rightarrow l^{n+1} = 0 \oplus l^n = l^n$
  $x_{n+1} = 1 \Rightarrow l^{n+1} = 1 \oplus l^n = \neg l^n$
  Individuiamo, concludendo la dimostrazione, la partizione cercata $(\mathcal{M}',\mathcal{N}')$ con:
  $\mathcal{M}' = M \cdot 0 \cup N \cdot 1$ e $\mathcal{N}' = M \cdot 1 \cup N \cdot 0$.

□

Questo significa che, qualsiasi siano i coefficienti della nostra FBL, a patto che questi non siano tutti 0, produrrano una funzione bilanciata.

Giunti a questo punto è lecito domandarsi se esistano altre (logicamente diverse) FBB non lineari e se sì, quante se ne possono ancora trovare? Il numero di FBL bilanciate per $n$ fissato, equivale al numero di *disposizioni con ripetizione di classe n* (coefficienti) da un insieme di due elementi $\{0, 1\}$ meno la disposizione $M^* = \overbrace{(0, 0, \cdots, 0)}^{n-volte}$ che identifica la costante 0, cioè $\mathbb{D}_{2,n} - 1 = 2^n - 1$. Tale numero è chiaramente inferiore al numero di FBB : $\mathbb{C}_{2^n,2^{n-1}}$ ricavato dal Corollario 2.2. E quindi sì, esisteranno sicuramente, per ogni $n$, $\mathbb{C}_{2^n,2^{n-1}} - (\mathbb{D}_{2,n} - 1)$ FBB non lineari.

### 2.2.2 Funzioni Booleane non Lineari Bilanciate

L'approfondimento delle proprietà e della teoria riguardante il vasto campo di studi delle funzioni Booleane, per quanto interessante, esula dai fini della tesi in oggetto. Rimanendo in tema, tuttavia, è bene accennare che la ricerca, specie per quanto riguarda l'indagine sulle funzioni Booleane bilanciate è molto attiva e di fondamentale importanza in applicazioni come la crittografia a chiave simmetrica [4].

Per rendere più realistica e articolata l'implementazione degli script presentati nel Capitolo 5, si è optato per la ricerca in letteratura di funzioni Booleane bilanciate non lineari da utilizzare nell'algoritmo di Deutsch-Jozsa. In particolare, è stata scelta una funzione a 5 variabili, ricavata con il metodo generale esposto da *Logachev* [5]:

$$h(x_1, x_2, x_3, x_4, x_5) = (x_1 \wedge x_2) \oplus x_3 \oplus (x_2 \wedge x_3 \wedge x_4) \oplus (x_2 \wedge x_3 \wedge x_5) \oplus (x_3 \wedge x_4) \oplus (x_4 \wedge x_5)$$

| $a$ | $b$ | $c$ | $d$ | $e$ | $c \veebar (a \wedge b) \veebar (c \wedge d) \veebar (d \wedge e) \veebar (b \wedge c \wedge d) \veebar (b \wedge c \wedge e)$ |
|---|---|---|---|---|---|
| T | T | T | T | T | F |
| T | T | T | T | F | F |
| T | T | T | F | T | T |
| T | T | T | F | F | F |
| T | T | F | T | T | F |
| T | T | F | T | F | T |
| T | T | F | F | T | T |
| T | T | F | F | F | T |
| T | F | T | T | T | T |
| T | F | T | T | F | F |
| T | F | T | F | T | T |
| T | F | T | F | F | T |
| T | F | F | T | T | T |
| T | F | F | T | F | F |
| T | F | F | F | T | F |
| T | F | F | F | F | F |
| F | T | T | T | T | T |
| F | T | T | T | F | T |
| F | T | T | F | T | F |
| F | T | T | F | F | T |
| F | T | F | T | T | T |
| F | T | F | T | F | F |
| F | T | F | F | T | F |
| F | T | F | F | F | F |
| F | F | T | T | T | T |
| F | F | T | T | F | F |
| F | F | T | F | T | T |
| F | F | T | F | F | T |
| F | F | F | T | T | T |
| F | F | F | T | F | F |
| F | F | F | F | T | F |
| F | F | F | F | F | F |

Computed by Wolfram|Alpha

Tabella di verità per $h$

# Capitolo 3

# L'Informazione Quantistica

Questa tesi si sviluppa nel contesto dell'Informazione Quantistica. L'Informazione Quantistica è una nuova branca di studi della teoria dell'informazione che trova la sua collocazione nell'intersezione tra Fisica, Matematica e Informatica. La nascita di questo nuovo campo di studi è dovuta in gran parte alle intuizioni del celeberrimo fisico Richard Feynmann, il quale fu il primo a suggerire, in una sua lezione nel 1981 [6], che il tipo di computer ideale che fosse stato capace di simulare i fenomeni fisici in maniera efficiente sarebbe stato solo un computer che avesse risposto alle stesse leggi della meccanica quantistica. L'argomento da lui proposto faceva perno sulla natura esponenziale che il mondo subatomico presenta se osservato dal punto di vista macroscopico e che quindi, avrebbe portato, per un processo transitivo, qualsiasi computer classico a sperimentare un tempo altrettanto esponenziale per simularlo. Spinti da questa osservazione, numerosi ricercatori si cimentarono nello studio e nella formalizzazione di questo nuovo paradigma di calcolo, gettandone le basi teoriche: nel 1982, Paul Benioff, nell'articolo "Quantum mechanical models of Turing machines that dissipate no energy" [7], dimostrò come un sistema basato sulla meccanica quantistica avrebbe potuto modellare una macchina di Turing (MdT) senza dissipare energia (il concetto della conservazione dell'energia verrà ripreso e chiarito in seguito). In altre parole, questi dimostrò come la fisica dei quanti fosse sufficiente ad

esprimere un qualsiasi modello tradizionale di calcolo; ma poteva addirittura fare di meglio? All'incirca nello stesso periodo, il fisico David Deutsch rispose a tale quesito [12] introducendo prima la controparte quantica della MdT universale (QMdTU) e poi dimostrando che la QMdTU riusciva a compiere operazioni al di fuori della portata della MdTU, come generare in maniera genuina numeri random, performare calcoli paralleli su di un unico registro e simulare perfettamente in tempo polinomiale sistemi fisici a stati di dimensione finita, confermando così la visionaria congettura che Feynmann aveva sollevato solo pochi anni prima.

## 3.1 Principi della Meccanica Quantistica

Nel mondo microscopico, quando si scende sotto la soglia atomica, cominciano ad emergere fenomeni che sono difficilmente interpretabili utilizzando il solo senso comune. Le nozioni più elementari come la determinazione unica delle proprietà fisiche di un oggetto (velocità, posizione, ecc.) a cui siamo abituati fin da piccoli, non sono più le stesse e, postulati come il principio di località o il determinismo, cessano di essere veri. Ciò portò inizialmente eminenti fisici del tempo a guardare con scetticismo o addirittura a rifiutare la meccanica quantistica per via delle stranezze a cui essa portava; numerosi esperimenti condotti successivamente hanno tuttavia confermato con successo le previsioni di questa e, ad oggi, nella comunità scientifica vi è unanime consenso sulla veridicità fenomenica della meccanica quantistica.

Al fine di capire il calcolo quantistico è necessario per prima cosa acquisire familiarità con le nuove leggi fondamentali che governano il mondo infinitamente piccolo. Vengono qui esposte sinteticamente le caratteristiche peculiari della fisica quantistica e, di seguito in maniera più formale, il modo in cui influenzano l'informatica.

### 3.1.1 Dai Numeri Reali ai Numeri Complessi

La meccanica quantistica differisce dalla maggior parte delle branche della scienza per via del largo uso che fa dei numeri complessi. I numeri complessi vennero per la prima volta introdotti come una curiosità matematica: $i = \sqrt{-1}$, chiamata unità immaginaria, era la soluzione "immaginaria" postulata all'equazione $x^2 = -1$. Per tanto tempo il campo dei numeri complessi è rimasto confinato unicamente nel reame della matematica fino a quando, con lo studio sistematico delle funzioni d'onda e l'introduzione delle analisi di Fourier, ci si accorse che i numeri complessi erano la struttura ottimale per descrivere in maniera compatta un'onda. La meccanica quantistica fa largo uso delle funzioni d'onda.

**Definizione 3.1.** *Un numero complesso $c$ è un'espressione*

$$c = a + b \times i = a + bi$$

*dove $a$ e $b$ sono due numeri reali, $a$ è chiamata la parte reale di $c$, mentre $b$ è la parte immaginaria.*

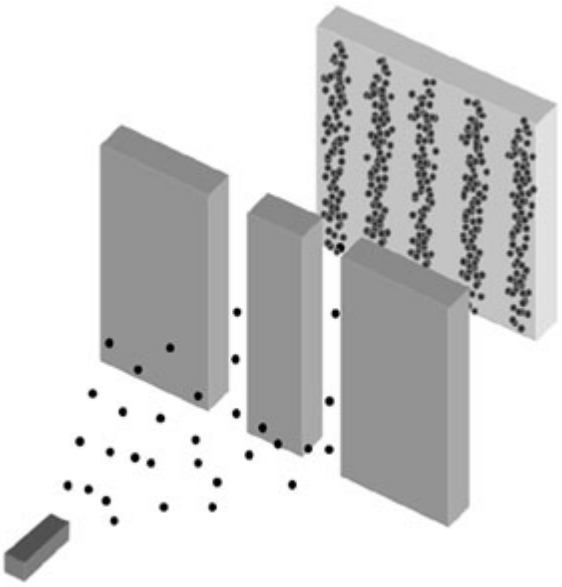
### 3.1.2 Da Stati Singoli alla Sovrapposizione degli Stati

Contrariamente a quello che abbiamo imparato fin da piccoli osservando il mondo in cui viviamo, un oggetto subatomico può trovarsi contemporaneamente in più posizioni differenti. Questo comportamento, detto di sovrapposizione, è tradizionalmente esemplificato riportando l'esperimento [13] delle due fenditure attraverso cui viene fatta passare, una alla volta, una qualsiasi particella che andrà a collidere contro un rilevatore posto dietro le fenditure. Contrariamente a quanto verrebbe da pensare, il rilevatore, non rileva come colpite dalle particelle solo le due aree in linea con le fenditure (e.g. come accadrebbe per dei palloni calciati in una porta coperta a meno di due sottoporte) bensì vengono sperimentalmente rilevate diverse aree di collisione, ognuna delle quali è colpita con un'intensità diversa, seguendo una distribuzione ad interferenza. Ogni singola particella discreta si comporta anche

come un'onda che, passando *contemporaneamente* per le due fessure, interferisce poi con se stessa. La dualità onda particella della materia scoperta grazie ai lavori di Luis de Broglie, Max Planck e altri fisici del secolo scorso, fu un risultato d'importanza cardinale per l'avvento della meccanica quantistica. Non solo la spazialità, ma anche altre proprietà come l'energia, il momento e lo spin di una particella sono soggette a sovrapposizione.

In realtà, non è possibile osservare direttamente il processo di sovrapposizione di una proprietà poichè, ogni qual volta viene effettuata una misurazione su di un sistema quantistico, questo verrà alterato inevitabilmente seguendo il principio di indeterminazione di Heisenberg e collasserà in uno ed uno solo degli stati sovrapposti, seguendo, come verrà chiarito più avanti, una ben definità distribuzione di probabiltà.

La sovrapposizione è il principale effetto che viene sfruttato per ottenere vantaggi computazionali.



visualizzazione dell'esperimento delle due fenditure

### 3.1.3 Dalla Località alla non Località

Centrale nella scienza moderna è la nozione secondo cui due oggetti, in un sistema, si influenzano tra di loro secondo un rapporto di causa-effetto solo se essi sono connessi da un'interazione fondamentale. La velocità di propagazione di una qualsiasi interazione è soggetta al limite imposto dalla velocità della luce nel vuoto, per cui non esisteranno due oggetti che, posti all'interno dello stesso sistema, entrino in relazione immediatamente. Questa assunzione, che prende il nome di principio di località, non è sempre valida nella fisica quantistica. È possibile infatti mettere due particelle in una relazione chiamata *entanglement* (le due particelle si diranno entangled) tale per cui non sia più possibile descriverne il comportamento in maniera separata, facendo si che un'operazione compiuta su di una avrà un effetto immediato sull'altra, indipendentemente dalla loro distanza.

### 3.1.4 Dalle leggi Deterministiche a quelle Non Deterministiche

Verso quale specifico stato collasserà una sovrapposizione di stati quando misurata? Mentre in altre aree della fisica le leggi sono deterministiche i.e. vi è un unico ouput se ripetiamo lo stesso esperimento più volte sotto le stesse condizioni, le leggi della meccanica quantistica ci dicono, tuttavia, che possiamo solamente conoscere la probabilità con cui un output, e non un altro, si verificherà.

Queste azioni, parafrasando Einstein stesso "terrificanti"[1] che emergono nel reame della meccanica quantistica, per quanto radicali ed avulse dalla nostra esperienza quotidiana, sono poi le stesse, riflettendo la natura esponenziale del mondo sub atomico, che conferiscono all'informazione quantistica il suo vero potenziale.

[1] "spooky action at distance" è la famosa frase che utilizzò per descrivere l'entanglement

## 3.2 I Sistemi a Confronto

Per vedere come la meccanica quantistica influenza l'informatica cominciamo presentando un modello per il calcolo che metta in luce le differenze quando applicato a più paradigmi differenti. In particolare vengono qui in ordine introdotti i concetti fondamentali del calcolo classico, del calcolo probabilistico e infine del calcolo quantistico.

### 3.2.1 Il Reame Classico

#### Il Bit

L'unità fondamentale su cui si fonda il calcolo classico è il bit. Consideriamo bit un qualsiasi sistema che espone un'unica proprietà osservabile che può assumere solo due valori. Chiamiamo questi 2 valori $|0\rangle$, $|1\rangle$[2]. Per essere coerentemente osservabili, $|0\rangle$ e $|1\rangle$ devono avere una precisa implementazione, ad esempio, se il bit è indicato da un transistor: $|0\rangle$ può essere identificato dalla presenza di un flusso di elettricità di un certo valore attraverso il transistor mentre $|1\rangle$ è identificato dall'assenza di corrente. Altre implementazioni che sfruttino la stessa logica sono ovviamente valide. Come già anticipato nel primo capitolo, $|0\rangle$ e $|1\rangle$ sono sufficienti per modellare una qualsiasi funzione booleana e, più in generale, un bit è l'unità fondamentale che governa ogni possibile operazione su un computer moderno.

#### Stati e Osservazioni

Una visione usuale dei sistemi di computazione, e dei sistemi in generale, è quella in cui viene specificato lo spazio dei possibili stati $\mathcal{B}_s$ in cui è possibile trovare il sistema in questione. Ogni sistema è inoltre specificato da una serie di proprietà chiamate *osservabili*, delle quali è possibile *misurare* il valore per ogni possibile stato appartenente allo spazio degli stati.

[2]cominciamo, senza alterarne il significato, ad utilizzare la notazione bra-ket introdotta dal fisico britannico Paul Dirac

Il sistema considerato è il bit. Ogni bit ha un solo osservabile dal valore $|0\rangle$ oppure $|1\rangle$ e, dal momento che questo è sufficiente a descrivere il comportamento del sistema, definiamo $\mathcal{B}_c$ lo spazio degli stati:

$$\mathcal{B}_c = \{|0\rangle, |1\rangle\}$$

per il bit classico.

Si noti come in questo specifico sistema lo spazio degli stati corrisponda esattamente con lo spazio dei possibili valori della proprietà osservabile infatti, se il bit si trova nello stato $|0\rangle$ allora la misura produrrà il valore $|0\rangle$, mentre se il bit si trova nello stato $|1\rangle$, la misura produrrà valore $|1\rangle$. Quest'affermazione può sembrare banale eppure vedremo presto come esistano sistemi dove l'equivalenza non sussiste e avremo bisogno di regole più complicate per determinare i valori che risultano da una misura.

**Trasformazioni**

Un sistema non è statico. È infatti sensato definire azioni che cambino lo stato attuale e quindi il risultato delle osservazioni future. Ad esempio, se un bit viene misurato a $|0\rangle$ e viene poi passato attraverso un invertitore (nella metafora del transistor ciò avviene bloccando il flusso di corrente), presenterà il valore $|1\rangle$ se osservato nuovamente. Formalmente, specifichiamo una tale *trasformazione* fornendo una funzione lineare $T : \mathcal{B}_s \to \mathcal{B}_s$ che agisce dallo spazio degli stati sullo spazio degli stati. Nell'esempio dell'invertitore tale funzione sarà uguale alla seguente applicazione lineare N:

$$N|0\rangle = |1\rangle, N|1\rangle = |0\rangle$$

oppure, un'altra possibile trasformazione può essere quella che pone lo stato del sistema a $|0\rangle$:

$$G|0\rangle = |0\rangle, G|1\rangle = |0\rangle$$

qualunque sia lo stato di partenza.

Ricapitolando, un generico bit ha un'unica proprietà fisica, quindi un unico osservabile. Una misura nel sistema risulta in $|0\rangle$ oppure in $|1\rangle$, le misure

non producono mai risultati intermedi. Una trasformazione può comportare un passaggio di stato nel sistema e, se il sistema viene misurato più volte senza che una trasformazione occorra tra le misure, si otterrà sempre lo stesso risultato.

### 3.2.2 Il Reame Probabilistico

#### Bit probabilistico

Si consideri ora un bit probabilistico. Un sistema che implementa un bit probabilistico può essere metaforicamente immaginato come il processo di un lancio di una moneta potenzialmente truccata. Ancora, questo sistema ha un unico osservabile, e cioè il risultato del lancio: $|0\rangle$ oppure $|1\rangle$. Il valore dell'osservazione dipende dallo sbilanciamento dello stato e cioè dal grado di manomissione della moneta. Più una moneta è truccata e più la probabilità di ottenere un risultato piuttosto che l'altro differisce dal 50%. Possiamo quindi esprimere uno stato del sistema come una funzione lineare sui valori dell'osservabile dove i coefficienti rappresentano la probabilità di osservare il valore.

### Stati e Osservazioni

Lo spazio degli stati corrisponderà a:

$$\mathcal{B}_p = \{a_0 |0\rangle + a_1 |1\rangle \ : a_0, a_1 \in \mathbb{R} \wedge a_0^2 + a_1^2 = 1\}$$

dove, per ogni stato, misureremo $|0\rangle$ con probabilità $a_0^2$ e $|1\rangle$ con probabilità $1 - a_0^2 = a_1^2$. La ragione per cui la probabilità equivale al quadrato del coefficiente verrà chiarita in seguito.

Si supponga una misurazione che porta a $|0\rangle$, lo stato attuale è quindi passato da uno opaco $a_0 |0\rangle + a_1 |1\rangle$ a un certo $1 |0\rangle + 0 |1\rangle$, quindi in questo sistema una misurazione altro non è che una trasformazione. Supponiamo inoltre come nel caso classico che misurazioni ulteriori non alterano lo stato già misurato: non è quindi possibile in alcun modo osservare direttamente la probabilità. Il bit probabilistico si trova in uno stato di incertezza solo nel momento antecedente al lancio.

### Trasformazioni

Sia T una trasformazione, se T agisce in un determinato modo su $|0\rangle$ e agisce in un determinato modo su $|1\rangle$, allora, essendo un'applicazione lineare sullo spazio degli stati, per un arbitrario bit probabilistico segue che:

$$\begin{aligned} T(a_0 |0\rangle + a_1 |1\rangle) =& T(a_0 |0\rangle) + T(a_1 |1\rangle) \\ =& a_0(T |0\rangle) + a_1(T |1\rangle) \end{aligned}$$

In questo modo è possibile con le giuste accortezze intuire i risultati di una trasformazione. Più nel dettaglio, la descrizione del comportamento del sistema è riscrivibile con l'algebra lineare. $\mathcal{B}_p$ è un sottoinsieme dello spazio vettoriale $\mathbb{R}^2$ e sia $\{|0\rangle, |1\rangle\}$ la base di tale insieme. Imponiamo che $\mathcal{B}_p$ possieda il prodotto scalare:

$$(a_0 |0\rangle + a_1 |1\rangle)(b_0 |0\rangle + b_1 |1\rangle) = (a_0 b_0) + (a_1 b_1)$$

che implica la norma Euclidea:

$$\|a_0 |0\rangle + a_1 |1\rangle\| = \sqrt{a_0^2 + a_1^2}$$

Allora, per definizione $\mathcal{B}_p$ è l'insieme di tutti i vettori con norma 1 appartenenti a $\mathbb{R}^2$. Si denoti con $\mathcal{P}_i$ la proiezione di $\mathcal{B}_p$ sull'i-esimo sottospazio:

$$\mathcal{P}_0(a_0 |0\rangle + a_1 |1\rangle) = a_0 |0\rangle$$
$$\mathcal{P}_1(a_0 |0\rangle + a_1 |1\rangle) = a_1 |1\rangle$$

Infine usiamo la notazione $prob[|\phi\rangle \rightarrow |i\rangle]$ per indicare la probabilità di ottenere $|i\rangle$ dalla misurazione di $|\phi\rangle$, che sarà:

$$prob[|\phi\rangle \rightarrow |i\rangle] = \|\mathcal{P}_i(|\phi\rangle)\|^2$$

. Dopo aver misurato $|i\rangle$, per esprimere la consistenza nelle future misurazioni dobbiamo assicurarci che lo stato risultante sia certo su $|i\rangle$, ciò è ottenuto normalizzando $\mathcal{P}_i(|\phi\rangle)$:

$$\frac{\mathcal{P}_i(|\phi\rangle)}{\|\mathcal{P}_i(|\phi\rangle)\|} = \frac{a|i\rangle}{\sqrt{a^2}} = 1 |i\rangle$$

. In generale, possiamo caratterizzare le trasformazioni per uno stato probabilistico come tutte quelle applicazioni lineari che preservano la norma.

Si noti come non è possibile disinguere tra gli stati $|\phi\rangle$ e $-|\phi\rangle$, in quanto la misurazione di questi produrrà valori che seguono la stessa distribuzione di probabilità e, l'applicazione di una qualsiasi trasformazione $T$, porta agli stati $T|\phi\rangle$ e $-T|\phi\rangle$ che sono a loro volta indistinguibili. La scelta di considerare tuttavia come distinti i due stati riflette alcune importanti proprietà, in particolare, nel caso quantistico queste saranno essenziali per esprimere gli effetti di interferenza dovuti alla natura ondulatoria del sistema.

### 3.2.3 Il Reame Quantico

#### Il Qubit

Ora che tutti i requisiti fondamentali sono stati presentati, è possibile introdurre il calcolo quantistico. Il sistema fondamentale in questo caso prende il nome di qubit. Analogalmente ai 2 casi precedenti, è utile astrarre dall'implementazione fisica e supporre che il sistema presenti un solo osservabile i cui possibili valori saranno i soliti $|0\rangle$ e $|1\rangle$[3].

#### Stati e Osservazioni

Lo spazio degli stati, in maniera simile a quanto visto nel caso probabilistico, è un sottoinsieme dello spazio vettoriale $\mathbb{C}^2$ tale che:

$$\mathcal{B}_q = \{c_0 |0\rangle + c_1 |1\rangle : c_0, c_1 \in \mathbb{C} \wedge |c_0|^2 + |c_1|^2 = 1\}$$

dove $||$ è il modulo del numero complesso calcolato come $|c| = |a + ib| = \sqrt{a^2 + b^2}$. Imponiamo anche in questo caso il prodotto interno su $\mathcal{B}_q$:

$$\langle \mathcal{C}, \mathcal{C}' \rangle = \langle c_0 |0\rangle + c_1 |1\rangle , c'_0 |0\rangle + c'_1 |1\rangle \rangle = (\overline{c_0} c'_0) + (\overline{c_1} c'_1)$$

Dove $\overline{c}$ indica la coniugazione di c. Vale quindi la seguente norma Euclidea:

$$\|\mathcal{C}\| = \sqrt{|c_0|^2 + |c_1|^2}$$

. $\mathcal{B}_q$ è il sottoinsieme di $\mathbb{C}^2$ composto da tutti i vettori di norma 1.

Senza riportare le equazioni, come nel caso probabilistico specifichiamo una misura in termini di proiezioni sui sottospazi formati da $|0\rangle$ e $|1\rangle$ e la probabilità dell'osservazione è determinata da $|c_0|^2$ per $|0\rangle$ e $|c_1|^2$ per $|1\rangle$.

**Definizione 3.2.** *Ogni stato espresso come combinazione lineare della base computazionale* $\{|0\rangle , |1\rangle\}$ *che non presenti elementi nulli è detto essere uno stato di sovrapposizione.*

[3]nella pratica un qubit è implementato utilizzando un unico osservabile di una particella. Ogni particella ha tanti (e teoricamente infiniti) osservabili: spin, quantità di moto, momento ecc.

### Trasformazioni

Prima di poter intendere una trasformazione di un sistema quantistico è necessario introdurre una particolare classe di matrici, le matrici unitarie

**Definizione 3.3.** *Una matrice $U \in \mathbb{M}^{n\times n}$ è unitaria se e solo se:*

$$UU^\dagger = U^\dagger U = I$$

*dove $U^\dagger[i,j] = \overline{U[j,i]}$.*

Come mostrato nel primo capitolo, qualsiasi trasformazione lineare che agisce su di un unico spazio vettoriale è rappresentata da una matrice quadrata. Si consideri per esempio la matrice:

$$N = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

la quale altro non è che la trasformazione che realizza l'invertitore anticipato nel caso classico, infatti, sia $|\phi\rangle = [c_0, c_1]^T$ la rappresentazione vettoriale di uno stato generico $|\phi\rangle$, allora l'applicazione di N su questo risulterà in:

$$N|\phi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_0 \end{bmatrix}$$

un nuovo stato le cui probabilità $prob[|\phi\rangle \to |x\rangle] = 1 - prob[N|\phi\rangle \to |x\rangle]$ sono invertite, che si traduce, per gli stati certi, nella complementazione del valore dell'osservabile. La matrice N viene per questo convenzionalmente chiamata trasformazione NOT.

Ritornando al discorso principale, è facile vedere come la matrice NOT sia anche unitaria:

$$NOTNOT^\dagger = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = I$$

Per convincersi che le unitarie siano le matrici giuste per descrivere la dinamica di uno stato quantistico è sufficiente metterne in risalto alcune proprietà:

**Lemma 2.** *Sia $U$ una matrice unitaria tale che $U \in \mathbb{M}^{n\times n}$, allora vale l'equazione:*

$$\langle UV, UV^{'}\rangle = \langle V, V^{'}\rangle$$

$\forall V, V^{'} \in \mathbb{C}^n$.

*i.e. le matrici unitarie rispettano il prodotto interno di uno spazio vettoriale complesso.*

*Dimostrazione.*

$$\langle UV, UV^{'}\rangle = (UV)^{\dagger}(UV^{'}) = V^{\dagger}U^{\dagger}UV^{'} = V^{\dagger}IV^{'} = \langle V, V^{'}\rangle$$

□

**Corollario.** *Le matrici unitarie preservano la norma:*

$$\|UV\| = \sqrt{\langle UV, UV\rangle} = \sqrt{\langle V, V\rangle} = \|V\|$$

In particolare se $\|V\| = 1$, allora $\|UV\| = 1$. Questo dimostra come le matrici unitarie siano adatte a descrivere i cambi di stato in $\mathcal{B}_q$.

## 3.3 Reversibilità

In realtà, le trasformazioni unitarie hanno un'altra importante caratteristica che permette di distinguere ulteriormente la dinamica di un sistema quantistico dalla dinamica di un sistema classico, supponiamo infatti di voler applicare l'applicazione lineare UNO ad un generico bit $|\phi\rangle(|\phi\rangle = |0\rangle$ oppure $|\phi\rangle = |1\rangle)$

$$UNO = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \Rightarrow UNO\,|\phi\rangle = |1\rangle$$

se, dopo l'applicazione si chiedesse ad un nuovo osservatore che osserva il sistema per la prima volta di ricostruire lo stato di partenza $|\phi\rangle$ è facile intuire che questi non possa fornire una risposta certa in quanto sia per $|\phi\rangle = |0\rangle$ sia per $|\phi\rangle = |1\rangle$ il risultato di $UNO\,|\phi\rangle$ è sempre $|1\rangle$. Sarebbe infatti come, riprendendo la metafora del transistor, una volta osservato un transistor in

cui non passa la corrente chiedersi se questo abbia mai ricevuto corrente. Siamo in presenza di trasformazioni che causano una perdita d'informazione, questo fenomeno nel mondo reale si traduce in rilascio di calore ed è uno dei problemi di cui tener conto durante l'ingegnerizzazione di un calcolatore[4].

Eppure esistono anche casi in cui l'informazione viene conservata, per esempio, con la trasformazione NOT prima introdotta, è possibile risalire facilmente al valore dello stato in input essendo: $NOT\,|0\rangle = |1\rangle$ e $NOT\,|1\rangle = |0\rangle$. Ciò è dovuto formalmente dall'esistenza dell'operazione inversa $NOT^{-1}$ la quale, come visto prima, altro non è che $NOT^{\dagger}$; infatti vale l'implicazione:

$$NOT\,|\phi\rangle = |\psi\rangle \Rightarrow NOT^{-1}NOT\,|\phi\rangle = NOT^{\dagger}NOT\,|\phi\rangle = NOT^{\dagger}\,|\psi\rangle = |\phi\rangle$$

dove $|\phi\rangle$ è un generico stato in input e $|\psi\rangle$ l'output della trasformazione. $NOT^{\dagger}$ permette in un certo senso di riavvolgere il calcolo, si dice in questo caso che la trasformazione è *reversibile*.

Alla luce di quanto è stato argomentato, ne consegue direttamente che: tutte le applicazioni unitarie sono reversibili, basta infatti sostituire all'equazione precedente NOT con una generica U.

**Lemma 3.** *Siano $U$ e $U'$ due matrici unitarie della stessa dimensione, allora il prodotto $UU'$ è una matrice unitaria.*

*Dimostrazione.*

$$(UU')(UU')^{\dagger} = (UU')(U'^{\dagger}U^{\dagger}) = UIU^{\dagger} = UU^{\dagger} = I$$

□

**Corollario.** *Sia $U_f = U_1U_2\cdots U_n$ la computazione per un qubit descritta da $n$ trasformazioni unitarie , allora $U_f$ è reversibile.*

Una computazione quantistica non dissipa energia a seguito di perdita dell'informazione, in quanto questa non avviene mai.

[4]È risaputo che per ogni bit d'informazione perso viene dissipata una quantita di energia $E = kT\ln 2$ dove $k = 1.3805 \times 10^{-23}$ è la costante di Boltzman

## 3.4 Un Esempio Quantistico

Un'altra trasformazione unitaria di fondamentale importanza è la matrice di Hadamard:

$$H = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

anche chiamata "fair coin split" per via del suo comportamento in grado di porre uno stato certo in uno stato di perfetta sovrapposizione $H|0\rangle = 1/\sqrt{2}|0\rangle + 1/\sqrt{2}|1\rangle$, $H|0\rangle = 1/\sqrt{2}|0\rangle - 1/\sqrt{2}|1\rangle$ che presenta una distribuzione di probabilità uniforme sui valori dell'osservabile. La matrice di Hadamard è l'inversa di se stessa:

$$\begin{aligned} H^2 &= \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ &= I \end{aligned}$$

Consideriamo i due stati: $|X\rangle = H|0\rangle$ e $|Y\rangle = H|1\rangle$, nonostante, come visto, non si ha perdita d'informazione durante i cambi di stato in $\mathcal{B}_q$, una volta applicata H, X e Y diventano due stati indistinguibili in termini di misurazioni, infatti $prob[|X\rangle \to |0\rangle] = prob[|Y\rangle \to |0\rangle] = \frac{1}{2}$ e lo stesso per il collasso su $|1\rangle$. I due sistemi, che prima risultavano chiaramente diversi sono ora uguali agli occhi di un osservatore. Eppure, sorprendentemente, riapplicare H porta a:

$$\begin{aligned} H|X\rangle =& H(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle) \\ =& \frac{1}{\sqrt{2}}(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle) + \frac{1}{\sqrt{2}}(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle) \\ =& 1|0\rangle + 0|1\rangle = |0\rangle \end{aligned}$$

e

$$\begin{aligned} H|Y\rangle =& H(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle) \\ =& \frac{1}{\sqrt{2}}(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle) + \frac{1}{\sqrt{2}}(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle) \\ =& 0|0\rangle + 1|1\rangle = |1\rangle \end{aligned}$$

che è la configurazione iniziale!

Durante l'evoluzione del sistema l'informazione non è quindi sempre accessibile in ugual modo bensì spesso questa viene mantenuta "nascosta" nell'ampiezza relativa di un componente dello stato: l'unica differenza tra X e Y, che non era misurabile direttamente, stava nel segno dell'ampiezza di $|1\rangle$. In altre parole, l'atto di misurazione comporta la perdita di un bit d'informazione (o $|0\rangle$ o $|1\rangle$) mentre, d'altra parte, riapplicare H e poi misurare consente di ricavare l'intero qubit. Si noti come non sia possibile ricreare lo stesso scenario con un sistema classico, lo stato è infatti sempre ben accessibile.

Per quanto possa sembrare limitativo, l'effetto esposto è invece centrale nella progettazione di molti algoritmi quantistici, infatti, l'applicazione di H consente di operare un cambiamento di base [8] sul sistema passando dalla base computazionale* alla base $\{\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\}$ (detta base di *Fourier*) di perfetta sovrapposizione in cui è possibile sfruttare l'informazione portata dalle ampiezze relative per operare calcoli utili utilizzando in questo modo il parallelismo offerto dalla sovrapposizione. I calcoli effettuati, si ricordi, non misurabili direttamente, vengono poi "estrapolati" riapplicando H e riportando il sistema nella base computazionale di partenza, seguendo un procedimento analogo a quello appena esposto. Questo potenziale computazionale cresce esponenzialmente all'aumentare della dimensione del sistema utilizzato.

## 3.5 Un Sistema a più Qubit

Si consideri ora un registro composto da $n$ bit. Ogni stato del registro così formato sarà rappresentabile dal vettore di qubit $|\mathbf{x}\rangle = |x_1\rangle |x_2\rangle \cdots |x_n\rangle$ che per brevità verrà spesso scritto come $|x_1 x_2 \cdots x_n\rangle$. Dal momento che ogni $|x_i\rangle$ può assumere i valori o $|0\rangle$ o $|1\rangle$, in totale, lo spazio degli stati del registro è composto da $\mathbb{D}_{2,n} = 2^n$ stati e viene indicato come:

$$\mathcal{B}_c^{(n)} = \left\{ \begin{array}{l} |0\rangle |0\rangle \cdots |0\rangle \\ |0\rangle |0\rangle \cdots |1\rangle \\ \cdots \\ |1\rangle |1\rangle \cdots |0\rangle \\ |1\rangle |1\rangle \cdots |1\rangle \end{array} \right\} = \Big\{ |\mathbf{x}\rangle = |x_1 x_2 \cdots x_n\rangle : |x_i\rangle \in \{0,1\} \Big\}$$

[5]. Così come $\mathcal{B}_q$ si sviluppa sopra la base computazionale $\{|0\rangle, |1\rangle\} = \mathcal{B}_c$, così allo stesso modo è possibile generalizzare un sistema quantistico sopra $\mathcal{B}_c^{(n)}$:

$$\mathcal{B}_q^{(n)} = \Big\{ \sum\nolimits_{x\in\{0,1\}^n} a_x |x\rangle : a_x \in \mathbb{C} \wedge \sum\nolimits_{x\in\{0,1\}^n} |a_x|^2 = 1 \Big\}$$

il cui spazio degli stati è composto da tutti i possibili vettori complessi di dimensione $2^n$ che hanno norma 1. Un sistema a $n$ qubit può trovarsi in una sovrapposizine di $2^n$ valori differenti, per esempio, il seguente stato

$$|\psi\rangle = \begin{array}{c} 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix} = \frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle$$

appartiene a $\mathcal{B}_q^{(2)}$. I numeri a fianco nella rappresentazione matriciale indicano la componente della base $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ a cui è associato il coefficiente complesso, si noti che $|\psi\rangle$ è uno stato di perfetta sovrapposizione.

[5] Nello specifico $\mathcal{B}_c^{(n)}$ è data dal prodotto tensoriale $\mathcal{B}_c \otimes \mathcal{B}_c \otimes \cdots \otimes \mathcal{B}_c$ di $n$ basi computazionali

Il sistema è composto da $n$ osservabili, uno per ogni qubit, le proiezioni $\mathcal{P}_0^{(i)}$, $\mathcal{P}_1^{(i)}$ sono formate da tutte le componenti dello stato nelle quali, rispettivamente, l'i-esimo qubit è $|0\rangle$ e l'i-esimo qubit è $|1\rangle$; la probabilità che l'i-esimo qubit venga misurato con valore $|x\rangle$ è calcolata di conseguenza: $prob[|\Psi\rangle \to |x_1 x_2 \cdot x \cdot \cdot x_n\rangle] = \left\|\mathcal{P}_x^{(i)}\right\|^2$, dove $|\Psi\rangle$ è uno stato generico. Il vettore risultante da una misurazione di un qubit è poi normalizzato analogamente a quanto già esposto per gli altri sistemi. Supponiamo ad esempio che venga misurato il secondo qubit di $|\psi\rangle$ (per convenzione rispetto alla rappresentazione di Dirac numeriamo i qubits da sinistra a destra) e che il valore osservato sia $|1\rangle$, allora:

$$\mathcal{P}_1^{(2)}(|\psi\rangle) = \frac{1}{2}|01\rangle + \frac{1}{2}|11\rangle$$

e lo stato $|\psi_1\rangle$ risultante dalla misurazione:

$$\begin{aligned} |\psi_1\rangle &= \frac{\mathcal{P}_1^{(2)}(|\psi\rangle)}{\left\|\mathcal{P}_1^{(2)}(|\psi\rangle)\right\|} = \frac{\frac{1}{2}|01\rangle + \frac{1}{2}|11\rangle}{\sqrt{(\frac{1}{2})^2 + (\frac{1}{2})^2}} \\ &= \frac{\frac{1}{2}|01\rangle + \frac{1}{2}|11\rangle}{\sqrt{\frac{1}{2}}} = \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{\sqrt{2}}|11\rangle \end{aligned}$$

è lo stato in cui il secondo qubit ha valore $|1\rangle$ con certezza mentre il primo si trova in perfetta sovrapposizione, con il 50% di probailità di misurare o $|0\rangle$ o $|1\rangle$, in accordo con la definizione di $|\psi\rangle$. In generale, $prob[|\Psi\rangle \to |\mathbf{x}\rangle] = |a_x|^2$.

### Trasformazioni

Le trasformazioni che descrivono la dinamica di questo registro sono le unitarie $\mathrm{U} \in \mathbb{M}^{2^n \times 2^n}$. In realtà, una qualsiasi trasformazione unitaria che agisca su di un singolo qubit del sistema risulta essere valida in quanto altera lo stato nel suo complesso, per convincersi di ciò, si supponga di applicare $NOT$ sul primo qubit di ${\mathcal{B}_c}^{(2)}$:

$$NOT_1 |00\rangle \rightarrow |10\rangle$$
$$NOT_1 |01\rangle \rightarrow |11\rangle$$
$$NOT_1 |10\rangle \rightarrow |00\rangle$$
$$NOT_1 |11\rangle \rightarrow |01\rangle$$

dove il pedice 1 è stato aggiunto per meglio esplicitare la semantica dell'operazione, come si può notare, questa trasformazione altro non fa che permutare gli elementi e, dal momento che ${\mathcal{B}_c}^{(2)}$ è la base su cui è costruito ${\mathcal{B}_q}^{(2)}$, l'azione di $NOT_1$ su di un generico stato:

$$NOT_1 |\Psi\rangle = NOT_1(\sum\nolimits_{x\in\{0,1\}^2} a_x |x\rangle) = \sum\nolimits_{x\in\{0,1\}^2} a_x NOT_1 |x\rangle$$

ne preserverà la norma. Più in generale, è possibile estendere una trasformazione $T : \mathcal{B}_q \rightarrow \mathcal{B}_q$ che agisce su di un singolo qubit in una trasformazione $T^{(n)} : {\mathcal{B}_q}^{(n)} \rightarrow {\mathcal{B}_q}^{(n)}$ che agisce su un intero registro quantistico.

Le trasformazioni su singoli qubits non sono le uniche trasformazioni ammissibili in un registro ma è possibile effettuare trasformazioni unitarie anche agendo su più qubits contemporaneamente, ad esempio, la porta[6] $C_iNOT_j$ ("controlled-NOT") è la trasformazione che effettua un NOT su l'j-esimo qubit se e solo se l'i-esimo qubit è $|1\rangle$ altrimenti lo lascia inalterato. Applichiamo $C_1NOT_2$ su ${\mathcal{B}_c}^{(2)}$:

$$C_1NOT_2 |00\rangle \rightarrow |00\rangle$$
$$C_1NOT_2 |01\rangle \rightarrow |01\rangle$$
$$C_1NOT_2 |10\rangle \rightarrow |11\rangle$$
$$C_1NOT_2 |11\rangle \rightarrow |10\rangle$$

ancora, $C_iNOT_j$ è unitaria su ${\mathcal{B}_q}^{(n)}$ perchè effettua una permutazione degli elementi di base. Tale trasformazione verrà ripresa per l'implementazione dell'algoritmo di Deutsch-Jozsa nell'ultimo capitolo.

[6]cominciamo a chiamare le trasformazioni con il nome "porta" in riferimento alla nomenclatura utilizzata nel modello a circuiti che verrà di seguito introdotto

## 3.6 Entanglement

Utilizzando le trasformazioni a più qubits come CNOT è possibile rendere il sistema entangled, è cioè possibile giungere in uno stato che non sia più esprimibile come composizione dei suoi sottosistemi. In questa situazione, le componenti sono intrinsecamente connesse e non possono essere descritte indipendentemente.

Si consideri l'evoluzione di un registro a due qubits partendo dallo stato iniziale:

$$|\psi_1\rangle = |00\rangle$$

applicando $H_1$ si ottiene:

$$|\psi_2\rangle = H_1 |\psi_1\rangle = \frac{|00\rangle + |10\rangle}{\sqrt{2}}$$

applicando ora $C_1NOT_2$ passiamo allo stato

$$|\psi_3\rangle = C_1NOT_2 |\psi_2\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

che è entangled. Infatti, se misuriamo il primo qubit ottenendo $|0\rangle$, allora sia avrà che $|\psi_3\rangle$ collasserà nello stato:

$$\begin{aligned} |\psi_4\rangle &= \frac{P_0^{(1)}(|\psi_3\rangle)}{\left\| P_0^{(1)}(|\psi_3\rangle) \right\|} = \frac{\frac{1}{\sqrt{2}} |00\rangle + 0 |01\rangle}{\sqrt{\frac{1}{2}}} \\ &= 1 |00\rangle + 0 |01\rangle = |00\rangle \end{aligned}$$

quindi, la misura del secondo qubit porterà a rilevare il valore $|0\rangle$ con certezza. Inversamente, se la misura avesse riportato $|1\rangle$ come valore del primo qubit, avremmo trovato $|1\rangle$ anche nel secondo sempre con certezza.

Da un punto di vista formale, un sistema si dice entangled se non può essere espresso come prodotto tensoriale delle sue componenti di base, nel caso di un sistema a due qubit, una generica composizione ha la forma:

$$(c_0 |0\rangle + c_1 |1\rangle) \otimes (c_0' |0\rangle + c_1' |1\rangle) = c_0c_0' |0\rangle |0\rangle + c_0c_1' |0\rangle |1\rangle + c_1c_0' |1\rangle |0\rangle + c_1c_1' |1\rangle |1\rangle$$

Il problema, sostituendo per $|\psi_3\rangle$, consiste nel risolvere il seguente sistema di equazioni:

$$\begin{cases} c_0 c'_0 = \frac{1}{\sqrt{2}} \\ c_0 c'_1 = 0 \\ c_1 c'_0 = 0 \\ c_1 c'_1 = \frac{1}{\sqrt{2}} \end{cases}$$

che, chiaramente, non ha soluzioni. Lo stato $|\psi_3\rangle$ non può essere riscritto come prodotto tensoriale degli stati base che sono quindi ora inseparabili.

Ritornando al significato fisico: in un certo senso, è come se nel preciso istante in cui viene compiuta un'osservazione, il valore calcolato venisse trasmesso *istantaneamente* al secondo qubit entangled il quale si comporta coerentemente con il valore trasmesso/osservato. Questa relazione tuttavia non contempla nè mezzi di trasmissione di questa ipotetica informazione, nè distanze tra i due sistemi. Sarebbe quindi possibile, almeno in linea di principio, utilizzare l'entanglement per comunicare subitaneamente anche a distanze superiori all'anno luce e ciò, come già anticipato, è in contraddizione con le leggi della relatività generale e con il principio di località su cui è fondata praticamente tutta la fisica non quantistica [9]. La correttezza delle previsioni teoriche è stata dimostrata ripetutamente in laboratorio grazie ad una serie di esperimenti [10] e sono state inoltre proposte, già a partire dal secolo scorso, numerose interpretazioni della meccanica quantistica che giustificherebbero il perchè dell'entanglement [11].

## 3.7 Sfruttare il Parallelismo

Il parallelismo, nel calcolo quantistico, emerge con la sovrapposizione. Come già visto nella sezione sui sistemi a più qubits, sia T una qualsiasi trasformazione unitaria che agisce sulla base ${\mathcal{B}_c}^{(n)}$ allora, l'applicazione di T su di uno stato $|\Psi\rangle \in {\mathcal{B}_q}^{(n)}$ si estenderà linearmente:

$$T\left|\Psi\right\rangle = T(\sum\nolimits_{x\in\{0,1\}^n} a_x \left|x\right\rangle) = \sum\nolimits_{x\in\{0,1\}^n} a_x T\left|x\right\rangle$$

T agisce quindi contemporaneamente su tutti i $2^n$ stati $\left|x\right\rangle \in {\mathcal{B}_q}^{(n)}$.

Sappiamo però che una misura dell'intero sistema comporta il collasso inevitabile del registro in una specifica configurazione con probabilità $|a_x|^2$; a primo impatto sembrerebbe dunque che il parallelismo sia solo una prerogativa di quel noumeno imponderabile che è la sovrapposizione, della quale non si può avere un quadro deterministico rendendo quindi apparentemente impossibile ricavarne vantaggi computazionali. Scopo di questa sezione, e in particolare della tesi in generale da questo punto in poi, è mostrare che non è così. L'utilizzo di particolari stati di partenza uniti a cambi di base appropriati permettono infatti la progettazione di algoritmi quantistici in grado di risolvere svariate tipologie di problemi con efficienza maggiore rispetto agli equivalenti algoritmi classici.

Viene qui di seguito proposta una procedura che, per una generica funzione Booleana $f : \{0,1\} \to \{0,1\}$, permette di calcolare $f(0)$ e $f(1)$ contemporaneamente applicando $f$ una volta sola.

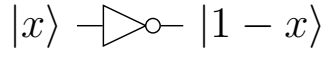
Per quanto dimostrato da Deutsch [12], il potere espressivo del calcolo quantistico è almeno equivalente a quello di un qualsiasi formalismo classico, è quindi possibile affermare che, per ogni funzione $g$ calcolabile esiste una corrispettiva procedura $U_g$ che la calcola. $U_g$ avrà la forma:

$$U_g\left|x,y\right\rangle = \left|x, y \oplus g(x)\right\rangle$$

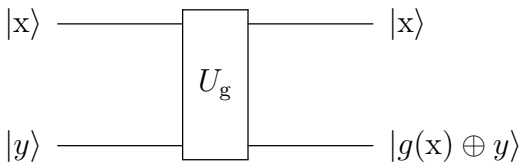
dove $\left|y\right\rangle$ è un qubit ausiliario che permette di rispettare il vincolo di reversibiltà: $U_g\left|x, y \oplus g(x)\right\rangle = \left|x, (y \oplus g(x)) \oplus g(x)\right\rangle = \left|x,y\right\rangle$ .

Per motivi di chiarezza espositiva e per rimanere fedeli all'approccio didattico utilizzato in letteratura per l'esposizione di questi argomenti affianchiamo al modello di calcolo fino a qui utilizzato il modello a circuiti. Il modello a circuiti è un'astrazione grafica di una procedura di calcolo che utilizza la metafora di un circuito elettronico, ogni trasformazione è raffigurata da una porta (detta logica nel caso classico e quantistica altrimenti) e l'evolu-

zione è da leggersi da sinistra verso destra. Come primi esempi consideriamo il circuito formato dalla porta NOT:




o il circuito che implementa $U_g$:

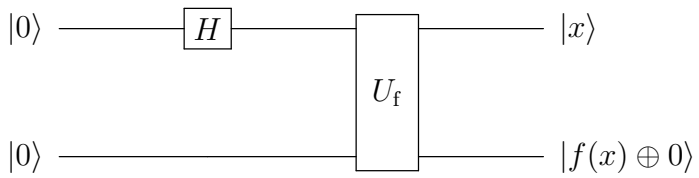



La semantica di questi diagrammi coincide con la semantica delle relative trasformazioni. In questo caso $U_g$ viene anche detto "oracolo" ad indicare una porta di cui non si conosce la struttura interna ma solo il mapping input e output.

Si ritorni allora al problema inizialmente sollevato riscrivendolo usando la terminologia introdotta: vogliamo calcolare di una funzione $f : \mathcal{B} \to \mathcal{B}$ se $f(0) = f(1)$ valutando $U_f$ una sola volta. È evidente che per risolvere classicamente il problema è necessario calcolare $f$ due volte, una per entrambi gli input.

Per sfruttare il parallelismo, il sistema deve essere posto in uno stato di sovrapposizione. Quale sia però la successione di trasformazioni che permetta di risolvere il problema è il vero ostacolo che si incontra nella progettazione di qualsivoglia algoritmo quantistico. Procediamo allora per tentativi al fine di indirizzarci verso la soluzione corretta. Dal momento che $U_f(|x, y\rangle) = |x, f(x) \oplus y\rangle$ calcola $f$ sul primo qubit, la scelta più intuitiva è quella di porlo in sovrapposizione.

Sia $|\psi_1\rangle = |00\rangle$ lo stato iniziale scelto arbitrariamente e



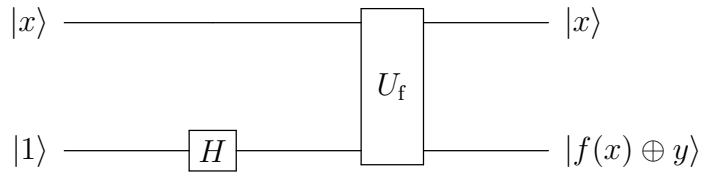
il circuito relativo. Per verificare se questo algoritmo (un circuito individua un algoritmo) sia sufficiente a risolvere il problema, monitoriamo l'evoluzione della computazione tramite $|\psi_i\rangle$ che descrive lo stato in cui si trova il registro dopo l'i-esima operazione. Cominciamo con l'applicare $H_1$, si avrà:

$$|\psi_2\rangle = H_1 |\psi_1\rangle = \Big[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\Big] |0\rangle = \Big[\frac{|00\rangle + |10\rangle}{\sqrt{2}}\Big]$$

e, dopo l'applicazione di $U_f$

$$\begin{aligned} |\psi_3\rangle = U_f |\psi_2\rangle &= \Big[\frac{|0, f(0) \oplus 0\rangle + |1, f(1) \oplus 0\rangle}{\sqrt{2}}\Big] \\ &= \Big[\frac{|0, f(0)\rangle + |1, f(1)\rangle}{\sqrt{2}}\Big] \end{aligned}$$

che calcola effettivamente $f(0)$ e $f(1)$! Solo che lo fa mantenendosi in sovrapposizione, un oggetto macroscopico come uno scienziato che non sperimenta gli effetti della meccanica quantistica può ugualmente ricavare $f(0)$ e $f(1)$ da questo circuito? La risposta è no, infatti, misurare uno qualsiasi dei 2 qubits permette soltanto d'ottenere informazioni su di un unico valore: o $f(0)$ con il 50% di probabilità o $f(1)$ con la stessa probabilità. È dunque necessario cambiare qualcosa, invece di porre il primo qubit in uno stato di sovrapposizione, si provi con il secondo settandolo inizialmente a $|1\rangle$. Per aumentare in generalità il primo qubit è lasciato in un generico $|x\rangle$. Ne risulta il seguente circuito:




l'evoluzione in questo caso è:

$$|\psi_2\rangle = H_2 |\psi_1\rangle = |x\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \left[\frac{|x,0\rangle - |x,1\rangle}{\sqrt{2}}\right]$$

di cui si noti il cambio di fase rispetto alla sovrapposizione nel tentativo precedente. Applicando poi $U_f$

$$|\psi_3\rangle = U_f |\psi_2\rangle = \left[\frac{|x, f(x)\oplus 0\rangle - |x, f(x)\oplus 1\rangle}{\sqrt{2}}\right] = |x\rangle \left[\frac{|f(x)\rangle - |\overline{f(x)}\rangle}{\sqrt{2}}\right]$$

$$\Rightarrow |\psi_3\rangle = \begin{cases} |x\rangle \left[\frac{|1\rangle - |0\rangle}{\sqrt{2}}\right] & f(x) = 1 \\ |x\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] & f(x) = 0 \end{cases} = (-1)^{f(x)} |x\rangle \left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$$

*Nota:* $a(b-c) = -a(c-b)$

ancora, misurare i 2 qubits non ci fornisce informazioni sulla struttura interna della funzione in quanto questa viene valutata solo sull'input $|x\rangle$ che non è mai posto in sovrapposizione. Tuttavia, questo tentativo non è stato vano, ci ha infatti permesso di esplicitare $f(x)$ come parametro per il primo stato, suggerendo così di mettere in sovrapposizione l'intero registro, questo passo permette di giungere alla soluzione cercata. Lasciamo inalterato il valore iniziale a $|1\rangle$ del secondo qubit e esplicitiamo $|x\rangle = |0\rangle$ (arbitrario, andrebbe bene anche $|1\rangle$), l'evoluzione sarà:

$$|\psi_2\rangle = H_1 |\psi_1\rangle = \left[\frac{|0,1\rangle + |1,1\rangle}{\sqrt{2}}\right]$$

,

$$|\psi_3\rangle = H_2 |\psi_2\rangle = \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] = \left[\frac{|00\rangle - |01\rangle + |10\rangle - |11\rangle}{2}\right]$$
$$= \frac{1}{2}[|0\rangle (|0\rangle - |1\rangle) + |1\rangle (|0\rangle - |1\rangle)]$$

. Una volta raggiunto uno stato di completa sovrapposizione degli osservabili e tenendo in mente il risultato del tentativo precedente, l'applicazione di $U_f$ porterà a:

$$|\psi_4\rangle = U_f |\psi_3\rangle = \frac{1}{2}[(-1)^{f(0)} |0\rangle (|0\rangle - |1\rangle) + (-1)^{f(1)} |1\rangle (|0\rangle - |1\rangle)]$$

il secondo qubit resta invariato, si può quindi semplificare

$$|\psi_4\rangle = \frac{1}{2}[[(-1)^{f(0)} |0\rangle + (-1)^{f(1)} |1\rangle][(|0\rangle - |1\rangle)]$$

a questo punto è possibile dimenticarsi del secondo qubit non dipendendo da $f$ e normalizzare lo stato. Andando poi a semplificare ulteriormente

$$|\psi_4\rangle_1 = \frac{1}{\sqrt{2}}[(-1)^{f(0)}(|0\rangle + (-1)^{f(1)\oplus f(0)} |1\rangle)]$$

finalmente $f(1)$ e $f(0)$ sono in relazione diretta. In particolare, si noti che: $f(1) \oplus f(0) = 0$ se e solo se $f$ costante, possiamo quindi riscrivere

$$|\psi_4\rangle_1 = \begin{cases} \pm[\frac{|0\rangle+|1\rangle}{\sqrt{2}}] & f\text{costante} \\ \pm[\frac{|0\rangle-|1\rangle}{\sqrt{2}}] & \text{altrimenti} \end{cases}$$

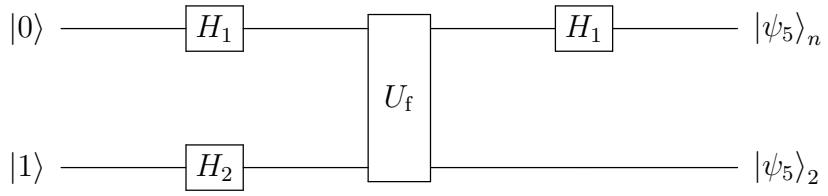
. Si noti come in realtà i segni $\pm$ individuano ogni possibile funzione $f : \mathcal{B} \to \mathcal{B}$ e non solo la proprietà di essere costante[7]. Questi valori non dovrebbero poi essere nuovi, difatti il primo qubit non è altro che $H_1 |0\rangle$ se $f$ costante e $H_1 |0\rangle$ altrimenti, inoltre, poichè H è la trasformazione inversa di se stessa, basterà riapplicarla per giungere finalmente a

$$|\psi_5\rangle = H_1 |\psi_4\rangle = \begin{cases} \pm |0\rangle [\frac{|0\rangle+|1\rangle}{\sqrt{2}}] & f \text{ costante} \\ \pm |1\rangle [\frac{|0\rangle-|1\rangle}{\sqrt{2}}] & \text{altrimenti} \end{cases}$$

. Per quanto l'informazione salvata nei segni andrà persa, non ci resta che misurare l'osservabile del primo qubit per risolvere il nostro problema: se risulta nello stato $|0\rangle$ allora $f(0) = f(1)$, mentre se risulta nello stato $|1\rangle$ allora $f(0) \neq f(1)$.

[7]Come visto nel primo capitolo sono $2^{(2^1)} = 4$ funzioni di cui 2 costanti e 2 no Ad esempio la $f'$ costante a 0 è rappresentata da $|\psi'_4\rangle_1 = +\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ essendo $(-1)^{f'(0)} = +1$.

Il procedimento appena seguito può essere riassunto dal circuito:




ed è lo stesso algoritmo che David Deutsch propose nel 1985 [12] dimostrando per la prima volta che l'Informazione quantistica poteva esserre impiegata per eseguire calcoli guadagnando in efficienza rispetto alla computazione classica.

È da sottolineare però che, nonostante la procedura esposta permetta di risolvere il problema in un'unica esecuzione invece che in due, il vantaggio ricavatone è solo costante e perciò non rilevante a livello asintotico. In termini di complessità, in realtà, è la formulazione di per sè a non essere interessante in quanto l'input di $f$ non è variabile ma fisso (0,1). Il problema quindi, non sembrerebbe essere il candidato ideale per scommettere sulla computazione quantistica, tuttavia, la soluzione di Deutsch fu (ulteriormente) importante poichè portò allo sviluppo del primo algoritmo in grado di sfruttare le vere potenzialità di questa disciplina.

# Capitolo 4

# L'Algoritmo di Deutsch-Jozsa

Il problema originale di Deutsch fu successivamente generalizzato da Deutsch stesso e Richard Jozsa nell'articolo "Rapid solution of problems by quantum computation" [15] nel 1992, il risultato che proposero fu poi rivisitato e completato da Richard Cleve, Artur Ekert, Chiara Macchiavello e Michele Mosca nel 1998 [16]. Questo lavoro è conosciuto in letteratura come l'algoritmo di Deutsch-Jozsa.

Se il primo algoritmo di Deutsch fu più un risultato simbolico, l'algoritmo di Deutsch-Jozsa fu invece il vero passo in avanti che mise in luce come, utilizzando il paradigma quantistico, si poteva ottenere un aumento esponenziale in velocità per la risoluzione di un determinato problema.

Nello specifico il problema diventa: *sia data una funzione* $f : \mathcal{B}^n \to \mathcal{B}$ *di cui non si conosce la definizione, determinare se tale funzione è costante oppure bilanciata*, sapendo per certo che* $f$ *sarà o costante o bilanciata.* D'ora in poi si farà riferimento a questo problema chiamandolo D.J.

Si noti come questa formulazione altro non sia che un'estensione della formulazione originaria, infatti, se nel caso particolare di $n = 1$ $f$ non è costante, allora sarà per forza bilanciata, e viceversa. Deutsch e Jozsa riuscirono a dimostrare che il problema poteva essere risolto per qualunque $n$ valutando $f$ una volta sola.

Viene di seguito mostrata la soluzione al problema mediante il sistema

classico e il sistema quantistico al fine di mostrare come quest'ultimo risulti vincente per complessità e correttezza.

**Teorema 4.1.** *Classicamente, il problema di D.J. appartiene a* $\Omega(2^{(n-1)})$*, cioè non esiste algoritmo che lo risolva con complessità* $o(2^{(n-1)})$*.*

*Dimostrazione.* Sia $f : \mathcal{B}^n \to \mathcal{B}$ espressa in un registro a $n$ bit attraverso la trasformazione $U_f : \mathcal{B}_c^{(n)} \to \mathcal{B}_c^{(n)}$ tale che $U_f |\mathbf{x}\rangle = U_f |00 \cdots f(\mathbf{x})\rangle$ sappiamo che $f$ è sicuramente o costante o bilanciata. Si noti che in questo sistema non è necessario imporre la reversibilità.

Una qualsiasi procedura che risolve il problema non può fare altro che cominciare a valutare $U_f$ su ogni possibile stato $|\mathbf{x}\rangle \in \mathcal{B}_c^{(n)}$ e fermarsi se:

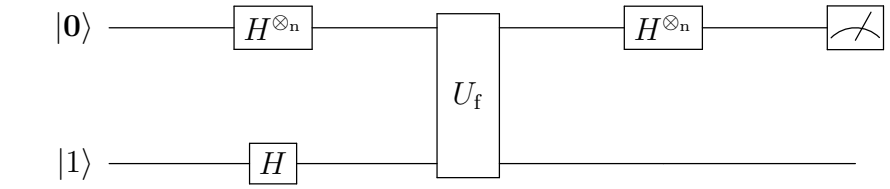
- Sono stati trovati due stati $|\mathbf{x}\rangle_1$, $|\mathbf{x}\rangle_2$ tali che $U_f(|\mathbf{x}\rangle_1) \neq U_f(|\mathbf{x}\rangle_2)$ poichè in questo caso $f$ sarà bilanciata non essendo costante;
- Sono stati già valutati $2^{(n-1)} + 1$ stati e tutti hanno effettuato lo stesso cambio di stato, poichè in questo caso $f$ sarà costante non essendo bilanciata.

Il caso pessimo coincide con quello costante e quindi ogni algoritmo che risolve D.J. nel caso classico avrà complessità $\Omega(2^{(n-1)})$. □

**Teorema 4.2.** *Un qualsiasi algoritmo probabilistico che compia* $k < 2^{n-1}+1$ *valutazioni di* $f(x)$ *di valori* $x \in \mathcal{B}^n$ *scelti indipendentemente e uniformemente, avrà almeno una probabilità di errore* $\epsilon \geq \frac{1}{2^{2k}}$ *[17].*

Questo risultato è ovviamente valido anche per un sistema a $n$ bit traducendo $f$ in $U_f$ come fatto per il teorema precedente. Utilizzando l'algoritmo probabilistico ottimale saranno comunque necessarie $2^{n-1} + 1$ valutazioni per rispondere con certezza e, una singola valutazione di $f$ (k=1) porterà a sbagliare mediamente 1 volta ogni 4.

L'algoritmo di Deutsch–Jozsa segue lo stesso schema dell'algoritmo di Deutsch solo generalizzandolo per un sistema a $n+1$ qubit. Il circuito che ne risulta è il seguente:




indica dove viene effettuata la misurazione finale.

Per mettere $n$ qubits in sovrapposizione è necessario applicare Hadamard ad ogni qubit, ciò è esprimibile sinteticamente attraverso il prodotto tensoriale $H^{\otimes n}$ di $n$ Hadamard. Si ricordi infatti che la composizione di più sistemi quantistici è data dal prodotto tensoriale degli stessi, lo stesso principio si applica alla composizione di trasformazioni. Per capire come costruire $H^{\otimes n}$ cominciamo ricavando $H^{\otimes 2}$ e cerchiamo di dedurne una struttura generale. Possiamo esplicitare $H$ come:

$$H = \frac{1}{\sqrt{2}} \begin{matrix} \\ 0 \\ 1 \end{matrix} \begin{matrix} \begin{matrix} 0 & 1 \end{matrix} \\ \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \end{matrix} = \frac{1}{\sqrt{2}} \begin{matrix} \\ 0 \\ 1 \end{matrix} \begin{matrix} \begin{matrix} 0 & 1 \end{matrix} \\ \begin{bmatrix} (-1)^{0\wedge 0} & (-1)^{0\wedge 1} \\ (-1)^{1\wedge 0} & (-1)^{1\wedge 1} \end{bmatrix} \end{matrix}$$

dove $H[i,j]$ indica l'azione di H sullo stato $|i\rangle$ dato lo stato $|j\rangle$ come input. Possiamo quindi calcolare semplicemente la composizione di 2 Hadamard:

$$H^{\otimes_2} = \frac{1}{\sqrt{2}} \begin{matrix} \\ 0 \\ 1 \end{matrix} \begin{matrix} \begin{matrix} 0 & 1 \end{matrix} \\ \begin{bmatrix} (-1)^{0\wedge 0} & (-1)^{0\wedge 1} \\ (-1)^{1\wedge 0} & (-1)^{1\wedge 1} \end{bmatrix} \end{matrix} \otimes \frac{1}{\sqrt{2}} \begin{matrix} \\ 0 \\ 1 \end{matrix} \begin{matrix} \begin{matrix} 0 & 1 \end{matrix} \\ \begin{bmatrix} (-1)^{0\wedge 0} & (-1)^{0\wedge 1} \\ (-1)^{1\wedge 0} & (-1)^{1\wedge 1} \end{bmatrix} \end{matrix} =$$

$$= \frac{1}{2} \begin{matrix} \\ 00 \\ 01 \\ 10 \\ 11 \end{matrix} \begin{matrix} \begin{matrix} 00 & 01 & 10 & 11 \end{matrix} \\ \begin{bmatrix} (-1)^{0\wedge 0} * (-1)^{0\wedge 0} & (-1)^{0\wedge 0} * (-1)^{0\wedge 1} & (-1)^{0\wedge 1} * (-1)^{0\wedge 0} & (-1)^{0\wedge 1} * (-1)^{0\wedge 1} \\ (-1)^{0\wedge 0} * (-1)^{1\wedge 0} & (-1)^{0\wedge 0} * (-1)^{1\wedge 1} & (-1)^{0\wedge 1} * (-1)^{1\wedge 0} & (-1)^{0\wedge 1} * (-1)^{1\wedge 1} \\ (-1)^{1\wedge 0} * (-1)^{0\wedge 0} & (-1)^{1\wedge 0} * (-1)^{0\wedge 1} & (-1)^{1\wedge 1} * (-1)^{0\wedge 0} & (-1)^{1\wedge 1} * (-1)^{0\wedge 1} \\ (-1)^{1\wedge 0} * (-1)^{1\wedge 0} & (-1)^{1\wedge 0} * (-1)^{1\wedge 1} & (-1)^{1\wedge 1} * (-1)^{1\wedge 0} & (-1)^{1\wedge 1} * (-1)^{1\wedge 1} \end{bmatrix} \end{matrix}$$

*Nota:* $(-1)^x * (-1)^y = (-1)^{x+y} = (-1)^{x \oplus y}$

viene utile allora definire la funzione $< , > : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$

$$\begin{aligned} < \mathbf{x}, \mathbf{y} > &= < x_0 x_1 \cdots x_{\mathrm{n}-1},\ y_0 y_1 \cdots y_{\mathrm{n}-1} > \\ &= (x_0 \wedge y_0) \oplus (x_1 \wedge y_1) \oplus \cdots \oplus (x_{\mathrm{n}-1} \wedge y_{\mathrm{n}-1}) \end{aligned}$$

che ci permette di scrivere sinteticamente $H^{\otimes_{\mathrm{n}}}$

$$H^{\otimes_{\mathrm{n}}}[i,j] = \frac{1}{\sqrt{2^n}} (-1)^{<i,j>}$$

. Proviamo a capire il comportamento della matrice ricavata: sia $|\mathbf{y}\rangle \in \mathcal{B}_c^{(n)}$ uno stato certo, riscritto vettorialmente

$$|\mathbf{y}\rangle = \begin{matrix} 00\cdots 0 \\ 00\cdots 1 \\ \cdots \\ \mathbf{y} \\ \cdots \\ 11\cdots 1 \end{matrix} \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 1 \\ \cdots \\ 0 \end{bmatrix} \qquad \mathbf{y} \in \{0,1\}^n$$

avremo

$$H^{\otimes_n} |\mathbf{y}\rangle = H^{\otimes_n}[-, \mathbf{y}] = \frac{1}{\sqrt{2^n}} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} (-1)^{<\mathbf{x},\mathbf{y}>} |\mathbf{x}\rangle$$

$$= \begin{array}{c} \\ 00\cdots0 \\ 00\cdots1 \\ \cdots \\ \mathbf{y} \\ \cdots \\ 11\cdots1 \end{array} \begin{bmatrix} \\ (-1)^{<00\cdots0,\mathbf{y}>} \frac{1}{\sqrt{2^n}} \\ (-1)^{<00\cdots1,\mathbf{y}>} \frac{1}{\sqrt{2^n}} \\ \cdots \\ (-1)^{<\mathbf{y},\mathbf{y}>} \frac{1}{\sqrt{2^n}} \\ \cdots \\ (-1)^{<11\cdots1,\mathbf{y}>} \frac{1}{\sqrt{2^n}} \end{bmatrix}$$

che è uno stato di perfetta sovrapposizione sopra la base computazionale, come ci aspettavamo.

Acquisiti gli strumenti necessari è possibile ora procedere alla formulazione e dimostrazione del seguente teorema.

**Teorema 4.3.** *L'algoritmo di Deutsch–Jozsa è un algoritmo deterministico che risolve il problema D.J. con un'unica valutazione di $U_f$.*

*Dimostrazione.* La correttezza dell'algoritmo è deducibile per costruzione svolgendo l'algoritmo stesso.

Sia $|\psi_i\rangle$ lo stato del sistema dopo l'i-esima operazione. Monitoriamo quindi l'evoluzione del sistema operazione per operazione. Lo stato iniziale $|\psi_1\rangle$ è:

$$|\psi_1\rangle = |\mathbf{0}\rangle |1\rangle = |0_1\rangle |0_2\rangle \cdots |0_n\rangle |1\rangle$$

dopo l'applicazione di $H^{\otimes_n}$ su $|\mathbf{0}\rangle$ il sistema si troverà nello stato:

$$\begin{aligned} |\psi_2\rangle &= \Big[\frac{1}{\sqrt{2^n}} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} (-1)^{<\mathbf{x},\mathbf{0}>} |\mathbf{x}\rangle\Big] |1\rangle \\ &= \Big[\frac{1}{\sqrt{2^n}} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle\Big] |1\rangle \end{aligned}$$

*Nota:* $< \boldsymbol{x}, \boldsymbol{0} >= 0$

dopo $H_{n+1}$:

$$|\psi_3\rangle = \Big[\frac{1}{\sqrt{2^n}} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} |\mathbf{x}\rangle\Big] \Big[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\Big]$$

come per l'algoritmo di Deutsch, aver posto in sovrapposizione il qubit ausiliario ci permette di esplicitare $f$ sui primi $n$ stati dopo l'applicazione di $U_f$:

$$|\psi_4\rangle = \Big[\frac{1}{\sqrt{2^n}} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})} |\mathbf{x}\rangle\Big] \Big[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\Big]$$

a questo punto possiamo dimenticarci dell'ultimo qubit concentrandoci su quello che accade dopo la riapplicazione di $H^{\otimes n}$ : otteniamo una sovrapposizione degli stati già in sovrapposizione!

$$\begin{aligned} |\psi_5\rangle_n &= \frac{1}{2^n} \sum\nolimits_{\mathbf{z}\in\{0,1\}^n} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})} (-1)^{<\mathbf{x},\mathbf{z}>} |\mathbf{z}\rangle \\ &= \frac{1}{2^n} \sum\nolimits_{\mathbf{z}\in\{0,1\}^n} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})\oplus<\mathbf{x},\mathbf{z}>} |\mathbf{z}\rangle \end{aligned}$$

ora $|\psi_5\rangle_n$ viene misurato. Invece di controllare subito il risultato chiediamoci prima quale sia la probabilità di osservare lo stato $|\mathbf{0}\rangle$, cioè:

$$\begin{aligned} prob[|\psi_5\rangle_n \to |\mathbf{0}\rangle] &= \|\mathcal{P}_{\mathbf{0}}(|\psi_5\rangle_n)\|^2 \\ &= \left\| \frac{1}{2^n} \sum\nolimits_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})} |\mathbf{0}\rangle \right\|^2 \end{aligned}$$

quindi, la probabilità di collassare a $|\mathbf{0}\rangle$ è totalmente dipendente da $f(\mathbf{x})$, in particolare:

**se $f$ costante:** $\mathcal{P}_{\mathbf{0}}(|\psi_5\rangle_n) = \frac{1}{2^n} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})} |\mathbf{0}\rangle = \pm\frac{2^n}{2^n} |\mathbf{0}\rangle = \pm 1 |\mathbf{0}\rangle$
$\Rightarrow prob[|\psi_5\rangle_n \to |\mathbf{0}\rangle] = 1$

**se $f$ bilanciata:** $\mathcal{P}_{\mathbf{0}}(|\psi_5\rangle_n) = \frac{1}{2^n} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})} |\mathbf{0}\rangle = \frac{2^{n-1}}{2^n} |\mathbf{0}\rangle - \frac{2^{n-1}}{2^n} |\mathbf{0}\rangle = 0 |\mathbf{0}\rangle$
$\Rightarrow prob[|\psi_5\rangle_n \to |\mathbf{0}\rangle] = 0$

.Verrà quindi misurato $|\mathbf{0}\rangle$ se e solo se la funzione $f$ è costante, altrimenti, verrà misurato un qualsiasi altro $|\mathbf{x}\rangle$ se e solo se la funzione è bilanciata.

□

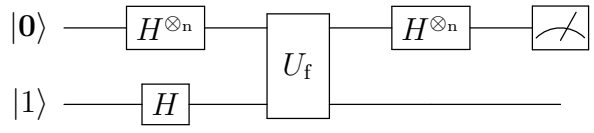
*In conclusione, l'algoritmo di Deutsch–Jozsa risolve il controverso problema con una sola esecuzione in contrasto alle* $2^{n-1}+1$ *necessarie per avere certezza nella computazione classica, risultando quindi esponenzialmente più veloce di ogni altra procedura conosciuta.*

# Capitolo 5

# Implementazione di Deutsch-Jozsa tramite QScript

Come ultimo argomento, questa tesi si propone di presentare un esempio concreto di quanto esposto fino ad ora. Viene di seguito implementato ed eseguito su più istanze l'algoritmo di Deutsch-Jozsa. Per sviluppare il lavoro è stata scelta la piattaforma web QuantumPlayground [18]: un esperimento di Chrome che simula il comportamento di un registro quantistico fino a 22 qubit programmabile attraverso il linguaggio di scripting ad hoc chiamato QScript. A supporto della programmazione QuantumPlayground fornisce un sistema di debugging e un motore grafico che permette di visualizzare l'evoluzione dello stato del sistema durante l'esecuzione.

L'obiettivo è quello di tradurre il circuito per D.J.




in un programma che ne simuli il comportamento. Per farlo, è necessario una funzione $f$ su cui valutare l'algoritmo, a tal proposito, sono state scelte due funzioni bilanciate e una costante. Di ogni funzione viene mostrato il procedimento seguito per la costruzione dell'oracolo, il codice relativo e uno snapshot del risultato finale.

Si cominci con il caso più elementare e cioè il caso in cui $f$ è costante. È stata scelta consapevolmente la $f$ costante a 0, il motivo di questa preferenza risiede nella definizione dell'oracolo, infatti, $U_f$ per $f(\mathbf{x}) = 0$ è semplicemente $U_f(|\mathbf{x}, y\rangle) = |\mathbf{x}, y \oplus f(\mathbf{x})\rangle = |\mathbf{x}, y\rangle$ ovvero la trasformazione identità.

Passiamo ad esporre e commentare lo script che implementa l'algoritmo per l'oracolo $I$, il primo passo è inizializzare il sistema:

```
//inizialization
VectorSize 8
SetViewMode 2
```

il primo comando crea un sistema di 8 qubits tutti inizializzati a $|0\rangle$. La scelta della dimensione è coerente con i successivi esempi ma fondamentalmente arbitraria per questo esempio. *SetViewMode* è una costante che setta la modalità di visualizzazione grafica, 2 indica la modalità $3D$. Per allinearsi con lo sato iniziale dell'algoritmo è necessario invertire l'ultimo qubit:

```
//swap the last qubit's value using X Pauli gate.
SigmaX 7
```

QScript ha come primitive le più comuni operazioni quantistiche tra cui, appunto, la porta X di Pauli ossia l'equivalente quantistica della trasformazione NOT. Come già esposto nel capitolo precedente, all'algoritmo sono necessarie due applicazioni di $H^{\otimes n}$, delle quali la prima in concomitanza con $H$ su $|1\rangle$; è utile allora definire una procedura che ponga in sovrapposizione un numero variabile di qubit:

```
//hadamard gate on j qubits, note: j=n on first call, j=n-1 on
    the scnd one
proc sysHgate j
    for i = 0; i < j ; i++
        Hadamard i
    endfor
endproc
```

. La procedura per l'oracolo sarà rappresentata in questo caso dalla procedura identità che è banalmente la funzione senza corpo:

```
//0 constant oracle
proc oracle

endproc
```

. Non rimane che codificare la misura dei primi 7 qubit. Anche in questo caso il linguaggio offre dei comandi "built-in" che si fanno carico di simulare il collasso dello stato osservato:

```
//we now measure the vector space
Measure

if measured_value == 0 //state 00000000
   Print "CONSTANT"
endif
if measured_value == 128 //state 10000000
   Print "CONSTANT"
endif

//if not constant, we know from assumption that must be balanced.
if measured_value != 0
   if measured_value != 128
      Print "BALANCED"
   endif
endif
```

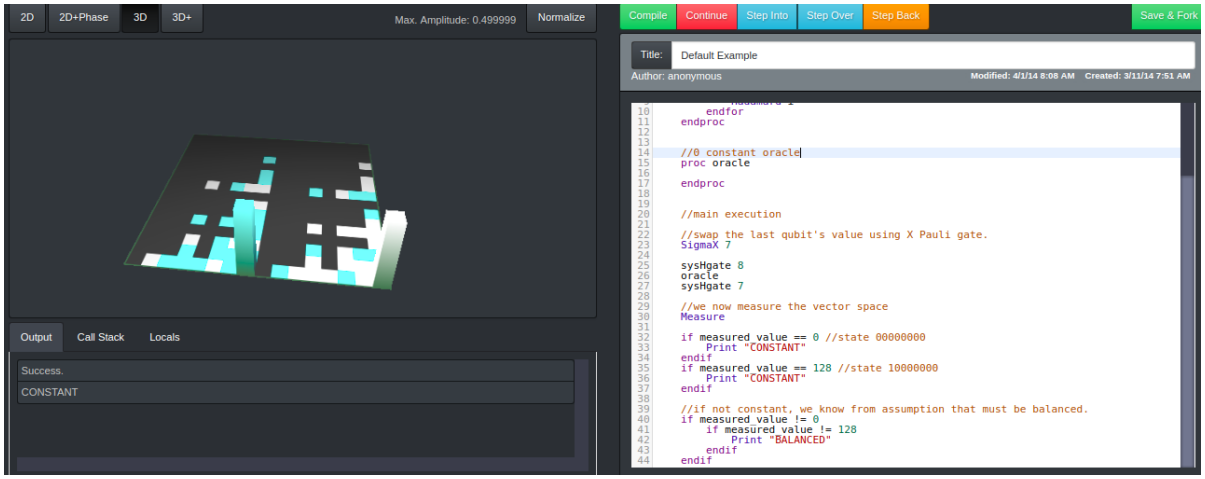
. Si noti come, dal momento che viene misurato l'intero registro, dobbiamo aggiungere dei controlli per tenere conto dei valori in cui può risultare il qubit ausiliario.

Definite le struttre principali, possiamo ora dare il codice completo:

```
//inizialization
VectorSize 8
SetViewMode 2


//hadamard gate on j qubits, note: j=n first call, j=n-1 on the
    second one
proc sysHgate j
    for i = 0; i < j ; i++
        Hadamard i
    endfor
endproc


//0 constant oracle
proc oracle

endproc


//main execution

//swap the last qubit's value using X Pauli gate.
SigmaX 7

sysHgate 8
oracle
sysHgate 7

//we now measure the vector space
Measure
```

```
if measured_value == 0 //state 00000000
    Print "CONSTANT"
endif
if measured_value == 128 //state 10000000
    Print "CONSTANT"
endif

//if not constant, we know from assumption that must be
    balanced.
if measured_value != 0
    if measured_value != 128
        Print "BALANCED"
    endif
endif
```

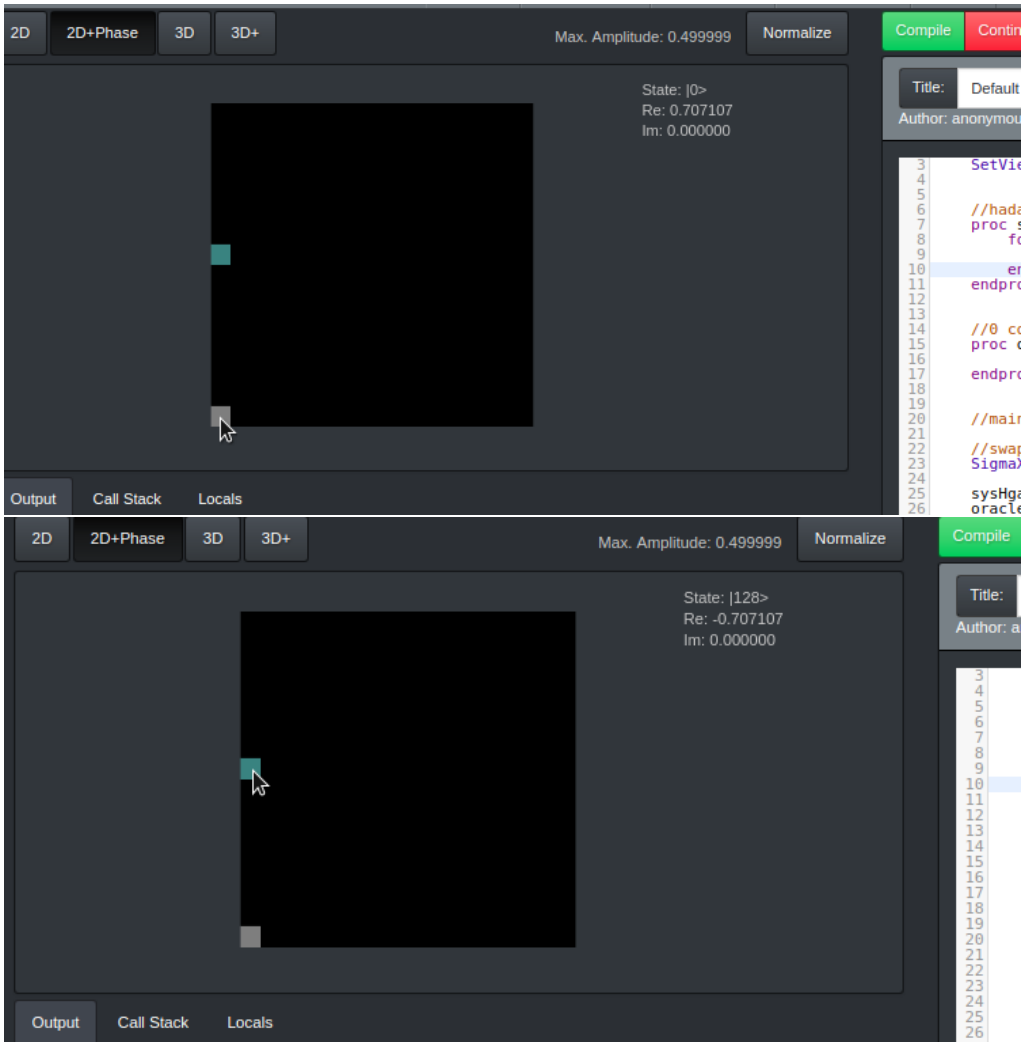
. Lo script è stato successivamente eseguito sul simulatore dimostrandosi corretto:




QuantumPlayground, Deutsch–Jozsa su funzione costante

. Si noti la visualizzazione grafica a sinistra che rappresenta lo stato del sistema subito prima della misurazione: il quadrato di base indica lo spazio degli stati per il registro e le aree colorate sono gli stati attualmente in

stato di sovrapposizione, rispettivamente, il bianco indica una fase positiva e il blu una negativa. La dimensione dell'area colorata sta invece ad indicare l'ampiezza dello stato sovrapposto, se è colorata solo la superfice allora l'ampiezza sarà 0 e può essere trascurata. Prima dell'osservazione il sistema si trova quindi in una sovrapposizione equiprobabile (deducibile dall'altezza dei rettangoli) di due stati con fasi opposte. Sopra il grafico a destra è anche segnata la probabilità dello stato che ha più possibilità di essere misurato che in questo caso è pari a $0.49999.. = 0.5$ che è coerente con quanto appena detto. Se, utilizzando un'altra modalità di visualizzazione, andiamo ad ispezionare quali sono questi due stati in sovrapposizione ci accorgiamo che:

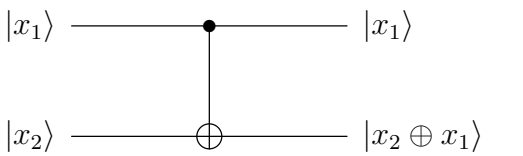


sono $|128\rangle = |10000000\rangle$ e $|0\rangle = |00000000\rangle$. La probabilità di misurare i primi 7 qubits a 0 è quindi certa. La funzione è quindi correttamente riconosciuta come costante.

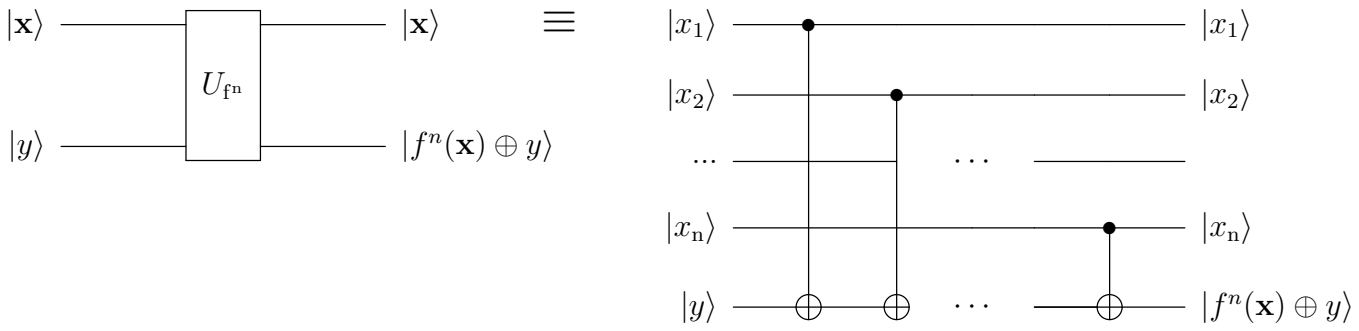
Passando alle funzioni bilanciate, come primo esempio si è scelta un'istanza della funzione bilanciata lineare $f^n$ definita nel primo capitolo. In generale l'oracolo $U_{f^n}$ vale:

$$U_{f^n}(|\mathbf{x}, y\rangle) = |\mathbf{x}, f^n(\mathbf{x}) \oplus y\rangle = |\mathbf{x}, x_1 \oplus x_2 \oplus \cdots \oplus x_n \oplus y\rangle$$

e chiaramente, questa volta non è possibile utilizzare la trasformazione identità bensì bisogna trovare una computazione che permetta di salvare sul qubit ausiliario $|y\rangle$ lo xor di tutti i qubit del sistema. Per risolvere il problema si è scelto di sfruttare la già introdotta trasformazione $CNOT$, la quale consente di operare uno xor tra due sistemi, infatti




dal momento che $|x_2\rangle$ viene invertito se e solo se $|x_1\rangle = |1\rangle$ la semantica sarà proprio quella dello $\oplus$. $U_{f^n}$ sarà quindi esprimibile con una concatenazione di $C_iNOT_{n+1}$, con $i$ distinti $1 \leq i \leq n$



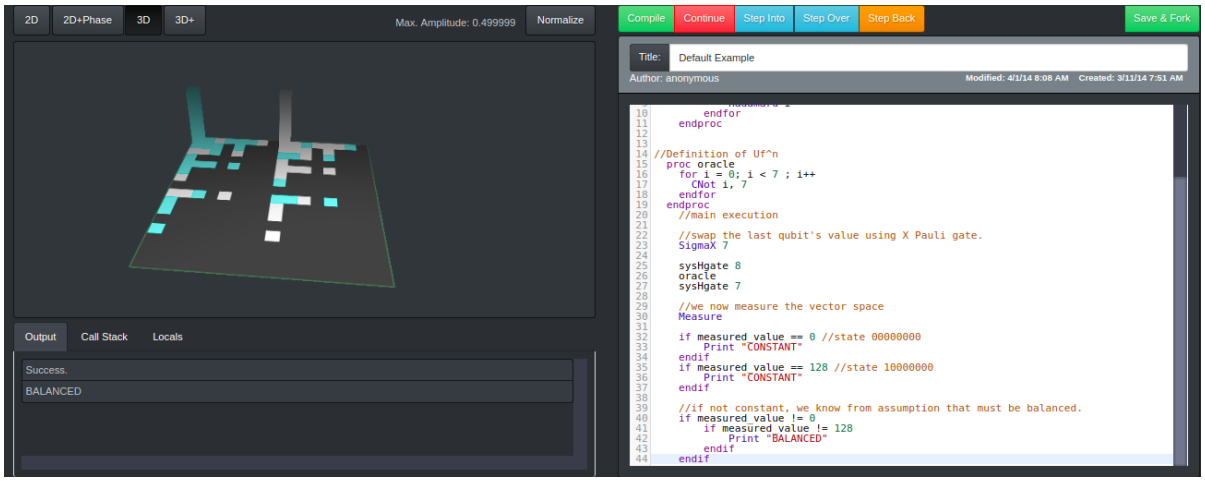

infatti: $(..(((x_1 \oplus y) \oplus x_2) \oplus x_3) \oplus .. \oplus x_n) = x_1 \oplus x_2 \oplus .. \oplus x_n \oplus y = f^n(\mathbf{x}) \oplus y$ .

Scegliendo sempre un registro a 8 qubits come sistema su cui calcolare $f^n$, l'unica parte che nel codice differisce dall'esempio precedente è appunto la procedura per $U_f$ che, per quanto argomentato, diventa:

```
//Definition of Uf^n
proc oracle
   for i = 0; i < 7 ; i++
      CNot i, 7
   endfor
endproc
```

si noti che QScript comincia a numerare da 0.

Il risultato dell'esecuzione conferma la correttezza dell'implementazione seguita




QuantumPlayground, Deutsch–Jozsa su funzione lineare bilanciata

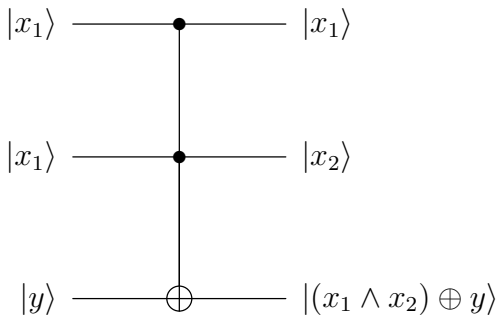
inversamente, anche in questo caso era possibile dedurre già dal grafico che la misurazione non avrebbe portato al collasso su $|\mathbf{0}\rangle$, la sovrapposizione infatti si trova nella zona diametralmente opposta rispetto a quella in cui si trovava per il caso costante. $|\mathbf{0}\rangle$ non è in sovrapposizone e quindi ha probabilità pari a 0 di essere osservato.

L'ultimo esempio vede l'implementazione della funzione non lineare bilanciata:

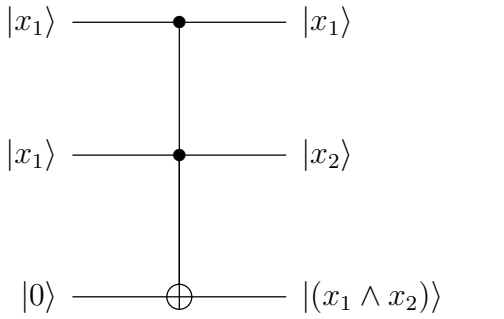
$$h(x_1, x_2, x_3, x_4, x_5) = (x_1 \wedge x_2) \oplus x_3 \oplus (x_2 \wedge x_3 \wedge x_4) \oplus (x_2 \wedge x_3 \wedge x_5) \oplus (x_3 \wedge x_4) \oplus (x_4 \wedge x_5)$$

anticipata nel primo capitolo.

Come si può intuire dalla definizione, per creare l'oracolo $U_h$, oltre ad utilizzare $CNOT$ per $\oplus$, bisogna trovare un metodo che ci permetta di calcolare la trasformazione $AND$ tra 2 e 3 stati. Si cominci col notare che l'$AND$ classico non può essere utilizzato in un circuito quantistico in quanto comporta la perdita di un bit di informazione andando contro al principio di reversibilità. L'approccio per rendere una funzione generica implementabile con una porta reversibile è stato tuttavia già esposto quando si è data la definizione di $U_f$, basta infatti aggiungere un qubit ausiliario $|y\rangle = |0\rangle$ e avremo l'AND quantistico $U_\wedge(|x_1, x_2, y\rangle) = |(x_1 \wedge x_2) \oplus y\rangle = |(x_1 \wedge x_2)\rangle$ il quale (lasciando $|y\rangle$ generico) altro non è che un CNOT esteso a 2 qubit di controllo: se e solo se i primi due qubit si trovano in $|1\rangle$ esegui NOT su $|y\rangle$. Questa operazione risulta essere di fondamentale importanza per la computazione quantistica [1] ed è conosciuta come la porta $Toffoli$, in nome dello scienziato che la scoprì. Viene rappresentata graficamente dal circuito

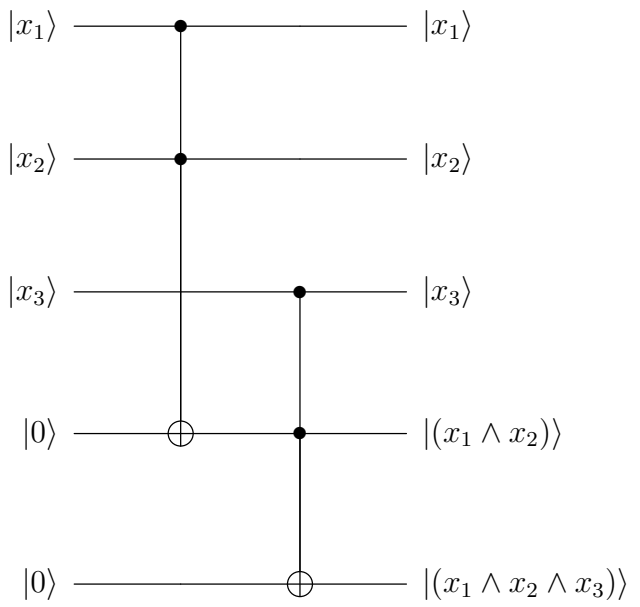



e, ponendo $|y\rangle = |0\rangle$

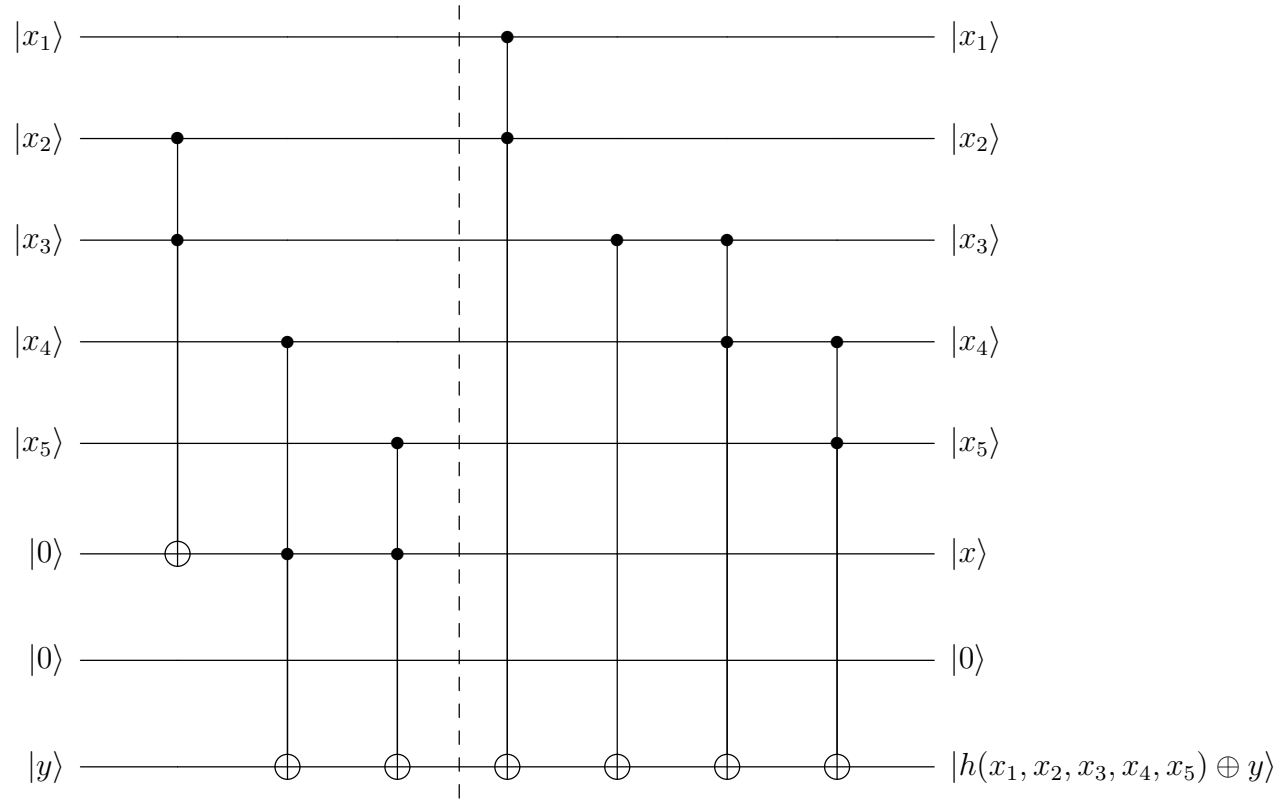



[1] La porta di Toffoli è universale per il calcolo logico reversibile [19] ed è "quasi" universale per la compuazione quantistica [20]

. E per quanto concerne un $AND$ a 3 stati? Per estensione verrebbe da concludere che un $CNOT$ a 3 qubit di controllo (o equivalentemente una $Toffoli$ a 3 qubit di controllo) con $NOT$ sul solito $|0\rangle$ sia sufficiente, tale risposta è corretta, tuttavia, QScript non fornisce una primitiva per $CNOT$ o $Toffoli$ generalizzati. Dobbiamo quindi cavarcela diversamente. Dopo un pò di ragionamento risulta relativamente facile capire che un $AND$ a 3 stati è ricavabile componendo due $Toffoli$, infatti




. Abbiamo adesso tutti gli strumenti necessari per costruire l'oracolo $U_h$ cercato. La costruzione viene divisa in due fasi: nella prima vengono calcolati tutti gli $AND$ a 3 stati e poi la parte rimanente della funzione. Come abbiamo visto, per implementare $AND$ a 3 stati abbiamo bisogno del supporto di un qubit a $|0\rangle$, dovremo quindi aggiungere al sistema tanti qubit quanti sono gli $AND$ distinti a 3 stati in $h$ (1). Ne risulta il seguente circuito a 7 qubit




si noti la presenza di un qubit ausiliario in più che non prende parte alla computazione, il motivo è dovuto ad una limitazione di QuantumPlayground che impone che la dimensione di un registro sia pari.

L'oracolo in QScript diventa

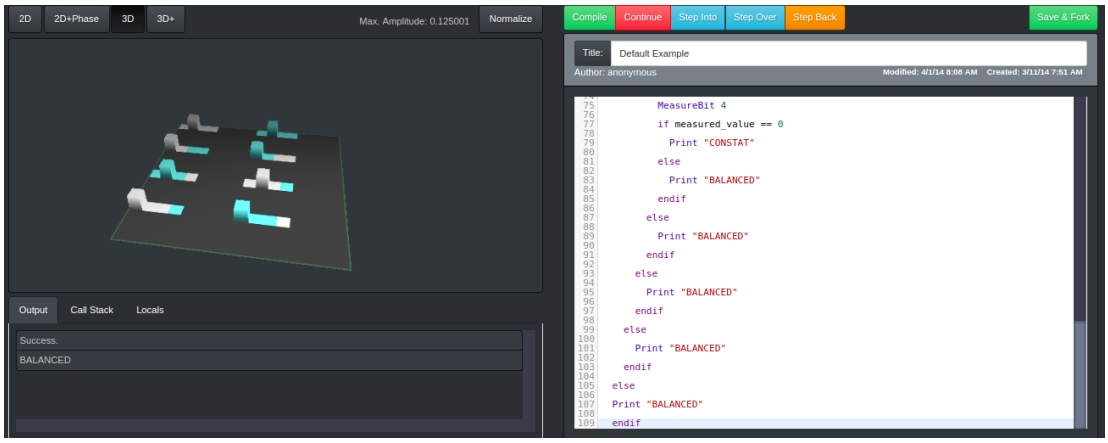
```
//Definition of Uh :h(x0,..,x4) xor x7
proc oracle
    Toffoli 1, 2, 5
    Toffoli 3, 5, 7 // x1 and x2 and x3
    Toffoli 4, 5, 7 // x1 and x2 and x4
    Toffoli 0, 1, 7 // x0 and x1
    Toffoli 2, 3, 7 // x2 and x3
    Toffoli 3, 4, 7 // x3 and x4
    CNot 2, 7 // x3
endproc
```

. Onde evitare di aggiungere svariati controlli per tutti i possibili valori dei qubit di supporto, viene in questo caso effettuata la misurazione solo su gli stati significativi:

```
MeasureBit 0

if measured_value == 0

   MeasureBit 1

   if measured_value == 0

      MeasureBit 2

      if measured_value == 0

         MeasureBit 3

         if measured_value == 0

            MeasureBit 4

            if measured_value == 0

               Print "CONSTAT"

            else

               Print "BALANCED"

            endif

         else
```

```
            Print "BALANCED"

         endif

      else

         Print "BALANCED"

      endif

   else

      Print "BALANCED"

   endif

else

Print "BALANCED"

endif
```

e, nonostante la verbosità del codice, anche in questo caso l'esecuzione restituisce il risultato sperato confermando la corretta implementazione dell'ultima delle funzioni testate.




QuantumPlayground, Deutsch–Jozsa su funzione non lineare bilanciata

# Capitolo 6

# Conclusioni

È stato provato come l'utilizzo della meccanica quantistica a supporto dell'informatica e in particolare nell'algoritmica permetta di trovare soluzioni ottimali. Riassumendo, l'idea di base per qualsiasi algoritmo quantistico è quella di utilizzare la sovrapposizione per lavorare con tutti i possibili input del problema contemporaneamente cercando, dal momento che questa sovrapposizione non è direttamente osservabile ma solo descrivibile come una distribuzione di probabilità, di trovare un pattern tra gli stati sovrapposti che sia in qualche modo collegato alla soluzione. Sfruttando poi i fenomeni di cancellazione tra stati dovuti all'interferenza, si tenta di estrapolare il più possibile tale pattern aumentando la probabilità di misurarlo. Non tutte le procedure ricavabili secondo questo schema sono deterministiche, molte infatti, come l'algoritmo di Shor, lavorano riducendo iterativamente la probabilità di sbagliare.

È importante menzionare il fatto che, contrariamente a quanto visto, esistono anche problemi in cui il nuovo paradigma è risaputo non portare vantaggi, come ad esempio i problemi indecidibili che rimangono tali anche per il calcolo quantistico[1], lasciando così inalterata la tesi di Church-Turing.

[1] è infatti possibile utilizzando un numero esponenziale di risorse simulare una macchina quantistica utilizzando una macchina di Turing, ne è un esempio proprio QuantumPlayground.

Le potenzialità esposte fino ad ora restano tuttavia dei validi motivi che giustificano la corsa alla costruzione di un computer quantistico che sta avvenendo in questi anni. Come la storia insegna, non è mai possibile prevedere interamente quali saranno i benefici e le scoperte dovute all'avvento di una nuova tecnologia, non resta quindi che continuare ad avanzare in un campo interdisciplinare così vasto alla ricerca di una maggiore comprensione dei misteri di un mondo così piccolo.

# Bibliografia


[1] Higham, N. (1998). *Handbook of writing for the mathematical sciences.* Philadelphia: SIAM, Soc. for Industrial and Applied Mathematics.

[2] Encyclopediaofmath.org. (2017). *Boolean function - Encyclopedia of Mathematics.* [online]

[3] Crama, Y. and Hammer, P. (2011). *Boolean Functions Theory, Algorithms and Applications.* 1st ed. Cambridge University Press, p.4.

[4] Dobbertin, Hans. (1994).*Construction of bent functions and balanced Boolean functions with high nonlinearity.* International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, 1994.

[5] Logachev, O. A. (2007).*On Perfectly Balanced Boolean Functions.* IACR Cryptology ePrint Archive 2007 : 22.

[6] Feynmann, R.P. (1982).*Simulating physics with computers.* International Journal of Theoretical Physics, 21(6/7):467-488 1982.

[7] Benioff, P. (1982).*Quantum mechanical models of Turing machines that dissipate no energy.* Physical Reviews Letterers, 48(23):1581-1585, 1982.

[8] Yanofsky, N. and Mannucci, M. (2013). *Quantum computing for computer scientists.* New York: Cambridge University Press, p.57.

[9] Einstein A., Podolsky B., Rosen N. (1935). *Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?.* Phys. Rev. 47 (10): 777–780.

[10] Hensen, B., Bernien, H., Dréau, A., Reiserer, A., Kalb, N., Blok, M., Ruitenberg, J., Vermeulen, R., Schouten, R., Abellán, C., Amaya, W., Pruneri, V., Mitchell, M., Markham, M., Twitchen, D., Elkouss, D., Wehner, S., Taminiau, T. and Hanson, R. (2015). *Loophole-free Bell inequality violation using electron spins separated by 1.3 kilometres.* Nature, 526(7575), pp.682-686.

[11] Schlosshauer, Maximilian; Kofler, Johannes; Zeilinger, Anton (2013). *A Snapshot of Foundational Attitudes Toward Quantum Mechanics.* Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics. 44 (3): 222–230

[12] Deutsch, David (July 1985). *Quantum theory, the Church-Turing principle and the universal quantum computer.* Proceedings of the Royal Society A. 400 (1818): 97–117

[13] Donati, O; Missiroli, G F; Pozzi, G (1973). *An Experiment on Electron Interference*. American Journal of Physics. 41: 639–644.

[14] Donald H Menzel, *Fundamental formulas of Physics*, volume 1, page 153;

[15] Deutsch, D., Jozsa, R. (1992). *Rapid solution of problems by quantum computation*. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences (Vol. 439, No. 1907, pp. 553-558).

[16] Cleve, R., Ekert, A., Macchiavello, C., Mosca, M. (1998). Quantum algorithms revisited. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences (Vol. 454, No. 1969, pp. 339-354).

[17] Bera, D. (2015). *A different Deutsch–Jozsa*. Quantum Information Processing, 14(6), 1777-1785.

[18] Quantum Playground. (2017). http://www.quantumplayground.net . Retrived 29 November 2017.

[19] Toffoli, T. (1980). *Reversible computing.* Automata, Languages and Programming, 632-644.

[20] Shi, Yaoyun (2003). *Both Toffoli and Controlled-NOT need little help to do universal quantum computation.* Quantum Information & Computation. 3 (1): 84–92.

[21] James, T. L. G. (1999). *Quantum computation: an Introduction.* A thesis for the bachelor degree, Harvard College.

[22] Rivest, R., Shamir, A., Adleman, L. (1978). *A method for obtaining digital signatures and public-key cryptosystems.* Communications of the ACM, 21(2), 120-126.

[23] Present, I. (2000). *Cramming more components onto integrated circuits.* Readings in computer architecture, 56.

[24] (2017) *IBM Builds Its Most Powerful Universal Quantum Computing Processors.* https://www-03.ibm.com/press/us/en/pressrelease/52403.wss#release. Retrieved 1 December 2017.

[25] Shor, P. (1999). *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer.* SIAM Review, 41(2), pp.303-332

[26] Bennett, C. H., Brassard, G. (2014). *QUANTUM CRYPTOGRAPHY: PUBLIC KEY DISTRIBUTION AND CON Tos5.*

[27] Stephen Jordan *Quantum Zoo Algorithms* Stephen Jordan's home page, http://math.nist.gov/quantum/zoo/. Retrived 1 December 2017.

[28] Grover, L. K. (1996). *A fast quantum mechanical algorithm for database search*. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (pp. 212-219). ACM.

[29] Ambainis, Andris (2014) *What Can We Do with a Quantum Computer* Institute for Advanced Study.

[30] Norton, Quinn (2007). *The Father of Quantum Computing*. Wired.com. Retrieved 1 December 2017.